HO
GENT

IT Fundamentals

Lesnota's

Lesgevers IT Fundamentals

Professionele Bachelor in de Toegepaste Informatica

Inhoudsopgave

HOOFDSTUK 1

Getallensystemen en Binair Rekenen

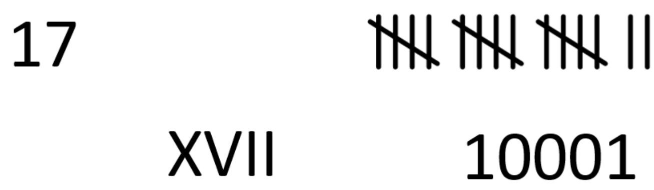
1.1 Inleiding

1.1.1 Voorstelling van getallen

Van oudsher werden er verschillende manieren gebruikt om getallen voor te stellen. In figuur 1.1 wordt het getal zeventien op verschillende manieren weergegeven.

Om aantallen te bepalen wordt dikwijls gebruik gemaakt van turven (zie figuur 1.1, bovenaan rechts): per eenheid trekt men een verticaal lijntje en indien men vier lijntjes heeft, en er komt nog een eenheid bij, doorhaalt men de vier verticale lijntjes met een schuine lijn. Op deze wijze kan men heel gemakkelijk de blokjes van vijf eenheden herkennen.

De Romeinen gebruikten een additief systeem, waarbij letters als symbolen gebruikt werden (zie figuur 1.1, links onderaan): hoofdletter i of dus I staat voor het getal één; hoofdletter v of V voor het getal vijf; hoofdletter x of X staat voor het getal tien enz. Door deze letters na elkaar te plaatsen maakt men de som van tien plus vijf plus één plus één, een totaal van zeventien. Vandaar de naam additief systeem.



Figuur 1.1: Voorstellingen van het getal 17

De twee andere voorstellingen maken gebruik van cijfers. Links bovenaan hebben we de tiendelige voorstelling en rechts onderaan de binaire voorstelling van het getal zeventien.

1.1.2 Definities

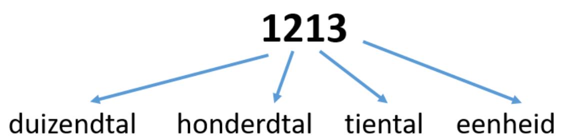
Definitie 1.1 *Een **cijfer** is een symbool, dat gebruikt wordt bij de voorstelling van getallen.*

Definitie 1.2 *Een **getal** wordt voorgesteld door, al dan niet van elkaar verschillende, cijfers achter elkaar te plaatsen.*

Voorbeeld:
Van in de lagere school hebben wij geleerd om getallen voor te stellen aan de hand van het tiendelig talstelsel. Dit houdt in dat we tien symbolen gebruiken, namelijk: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Door deze cijfers na elkaar te plaatsen krijgen we een getal: bijvoorbeeld 1213.

1.1.3 Positie van een cijfer

Beschouw het getal 1213. Eveneens in de lagere school hebben we geleerd dat het laatste cijfer, namelijk het cijfer 3 de eenheid, het voorlaatste cijfer het tiental, het derde laatste cijfer het honderdtal en het eerste cijfer het duizendtal voorstelt binnen het getal 1213 (zie figuur 1.2).



Figuur 1.2: Plaats eenheid, tiental, honderdtal, ... in een getal

We bemerken dat afhankelijk van de positie in het getal, een cijfer dus een andere betekenis heeft. Het is net die positie van het cijfer binnen het getal, dat aangeeft met hoeveel het cijfer vermenigvuldigd moet worden (zie figuur 1.3).

$$\begin{aligned} 1213 &= 1 \,.\, 1000 + 2 \,.\, 100 + 1 \,.\, 10 + 3 \,.\, 1 \\ &= 1 \,.\, 10^3 + 2 \,.\, 10^2 + 1 \,.\, 10^1 + 3 \,.\, 10^0 \end{aligned}$$

Figuur 1.3: Belang positie van de cijfers in het getal 1213

Zo wordt de eenheid vermenigvuldigd met 1 of 10 tot de macht 0, het tiental met 10 of 10 tot de macht 1, het honderdtal met 100 of met 10 tot de macht 2, het duizendtal met 1000 of 10 tot de macht 3, enz.

1.2 Positionele systemen

Definitie 1.3 *Een* ***positioneel (getallen)systeem*** *of* ***positioneel talstelsel (kortweg positiestelsel)*** *, is een talstelsel of getallensysteem, waarbij een getal wordt voorgesteld door een reeks van symbolen of cijfers. De positie of de plaats van het cijfer in het getal bepaalt de bijdrage aan het getal op basis van een gekozen grondtal.*

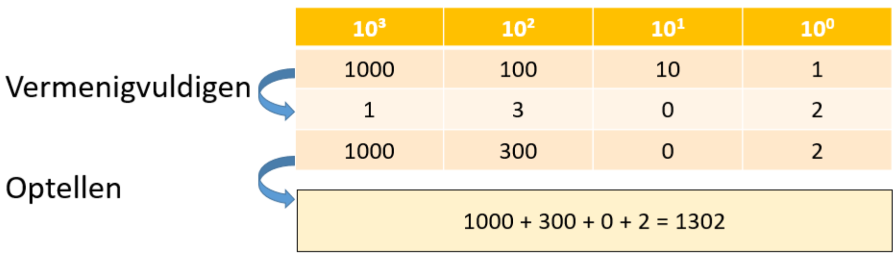
Algemeen: een positioneel systeem heeft een **grondtal a** en een **verzameling symbolen of cijfers**, waarvan het aantal gelijk is aan a.

1.2.1 Tiendelige getallen

De tiendelige getallen of het decimaal stelsel dat we, met ons allen, geleerd hebben in de lagere school is een positioneel systeem met als grondtal het getal 10 en als verzameling cijfers = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

De getallen worden gevormd door te vermenigvuldigen met de juiste macht van het grondtal, in casu het grondtal 10, afhankelijk van de positie van het cijfer.

Beschouw figuur 1.4 waar dit uitgewerkt is voor het getal 1302:



Figuur 1.4: Uitwerking voor het getal 1302

In de tabel in figuur 1.4 wordt in de bovenste rij de machten van 10 neergeschreven opgaand van rechts naar links; in de rij eronder of dus de tweede rij, worden deze machten uitgewerkt; in de derde rij worden de cijfers geplaatst van het getal, waarbij men de eenheden uiterst rechts plaatst en per grootheid één vak naar links opschuift. Vervolgens worden de cijfers uit de derde rij vermenigvuldigd met de getallen uit de tweede rij, en het resultaat hiervan wordt genoteerd in de onderste rij. Uiteindelijk worden alle resultaten uit de laatste rij bij elkaar opgeteld.

In figuur 1.5 worden de meest voorkomende decimale veelvouden weergegeven.

benaming	symbool	macht van 10	aantal eenheden
tera	T	10^{12}	1000 000 000 000
giga	G	10^{9}	1 000 000 000
mega	M	10^{6}	1 000 000
kilo	k	10^{3}	1 000
milli	m	10^{-3}	0,001
micro	µ	10^{-6}	0,000 001
nano	n	10^{-9}	0,000 000 001
pico	p	10^{-12}	0,000 000 000 001

Figuur 1.5: Meest voorkomende decimale veelvouden

1.2.2 Binaire getallen

1.2.2.1 Voorstelling

Het binair talstelsel of tweetallig talstelsel of, kortweg, de binaire getallen vormen een positioneel talstelsel met als **grondtal**, het getal **2** en als **verzameling cijfers = 0, 1**.

Voorbeeld van een binair getal = 10100010110.

De betekenis van elk symbool wordt weergegeven in figuur 1.6.

2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}
1	0	1	0	0	0	1	0	1	1	0

Figuur 1.6: Betekenis symbolen in binaire getallen

Wederom worden de machten van het grondtal, nu het getal 2, neergeschreven, opgaand van rechts naar links, met daaronder de cijfers van het getal op de overeenstemmende posities.

1.2.2.2 Toepassingsgebied

Binaire getallen worden gebruikt binnen de computerwereld om informatie voor te stellen en door te geven. Meestal worden de signalen overgebracht door middel van elektrische pulsen op twee verschillende voltages, waarvan er één overeenstemt met de digitale waarde 0 en de andere met 1.

In het bijzonder zullen we het **IPv4-adres** nader bekijken. Dit is een logisch adres dat een computer (eigenlijk de netwerkkaart) krijgt zodat communicatie met andere toestellen in een netwerk mogelijk is. Afhankelijk van het netwerk waarin de computer zich bevindt, zal dit een ander adres zijn. Een IPv4-adres bestaat uit 32 bits. Bijvoorbeeld : 11000000 10101000 00001010 00000001.
Bij netwerkproblemen, zal de administrator een IPv4-adres gebruiken om de oorzaak van de storing te achterhalen. Echter, zulk een opeenvolging van nulletjes en eentjes is voor het menselijk brein niet zo evident om te onthouden, laat staan om dit foutloos over te tikken. Er stelt zich, dus, een probleem doordat binaire getallen erg lang kunnen zijn en daarom soms niet overzichtelijk waardoor kansen op fouten reëel zijn.
Een computer kan perfect werken met enkel nullen en enen maar dit is niet zo voor het menselijk brein. Om het werkbaar te maken voor de gebruikers, bestaat de oplossing er in om de binaire getallen voor te stellen in een ander talstelsel.
Zo wordt bovenstaand IPv4 adres, in 4 groepen van 8 bits verdeeld. Elke groep van 8 bits wordt omgezet in de overeenkomstige decimale waarde (hoe dit gebeurt, wordt later besproken). De groepen van 8 bits worden gescheiden door een punt. Dit geeft als resultaat 192.168.10.1 .

1.2.2.3 Afspraken en notatie

Volgend probleem stelt zich: indien we het getal 10 schrijven, is dit dan het tiendelig getal 10 of het binair getal 10?

Beide getallen hebben niet dezelfde waarde maar wel **dezelfde notatie**. We moeten dus een systeem bedenken om onderscheid te maken tussen beide.

Daarom maken we, binnen deze cursus, gebruik van volgende notatie: we zetten haakjes rond het getal en noteren het grondtal als subscript (zie figuur 1.7).

$(10)_{10}$ = 10 als tiendelig getal
$(10)_2$ = 10 als binair getal

Figuur 1.7: Decimale en binaire 10

Om het niet te ingewikkeld te maken, vooral wanneer we continu werken met tiendelige getallen, maken we volgende afspraak: indien er geen notatie van het grondtal is, dan nemen we aan dat het getal in zijn decimale vorm staat.

1.2.2.4 Binair tellen

Bij het tellen in het decimaal stelsel, beginnen we met de eenheden te tellen in opgaande volgorde. Bij deze eenheden zijn de tientallen, honderdtallen, enz. gelijk aan 0.
We tellen dus : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Na 9 wordt het tiental met 1 verhoogd. Dus het cijfer van de rang één hoger (dan de eenheden) wordt vermeerderd met 1.
We laten dit tiental op 1 staan, waarbij de eenheden opnieuw de range van 0, 1, 2, 3, 4,

5, 6, 7, 8, 9 doorlopen. Opnieuw na 9, wordt het tiental met 1 verhoogd, dus krijgen we voor het tiental het cijfer 2. Opnieuw doorlopen de eenheden de range 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 en zo gaat dit verder tot wanneer de tientallen ook de waarde 9 krijgen. Na het getal 99 worden de honderdtallen met 1 verhoogd, dus opnieuw wordt het cijfer van de rang één hoger vermeerderd met 1. We beginnen dan opnieuw met de eenheden toe te voegen en daarna de tientallen op oplopende volgorde.

Belangrijk *is dat na het bereiken van het hoogste cijfer, de rang één hoger met het cijfer 1 verhoogd wordt.*

Binair tellen verloopt op een analoge manier als bij het decimaal tellen:

- Stap 1: Begin bij 0
- Stap 2: Vervang het laatste cijfer door zijn opvolger:
  - Stap 2.1: Indien het laatste cijfer 0 is wordt dit vervangen door 1
  - Stap 2.2: Wanneer een 1 (of dus het hoogste cijfer) verhoogd moet worden, verhoog dan het cijfer van één rang hoger met 1
- Stap 3: Ga verder met stap 2

Binaire waarden	Decimale waarden
0	0
1	1
10	2
11	3
100	4
101	5
110	6
111	7
1000	8
1001	9
1010	10
1011	11
1100	12
1101	13

Figuur 1.8: Binair en decimaal tellen

In figuur 1.8 wordt in de eerste kolom binair geteld startend vanaf 0. De overeenkomstige decimale waarde wordt weergegeven in de tweede kolom, waar ook het telproces bij de decimale getallen gedemonstreerd wordt.

1.2.2.5 Definities

Een aantal definities met betrekking tot binaire getallen, zijn hier aan de orde.

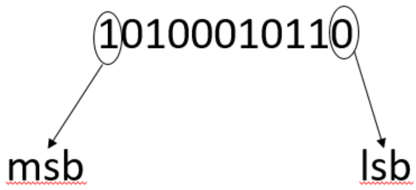
Definitie 1.4 *Een **bit of ook nog binary digit** , afgekort als b (kleine letter b), is één binair cijfer. Een bit kan dus maar de waarde 0 of 1 aannemen.*

Definitie 1.5 *Een **nibble** is een rij van 4 bits.*

Definitie 1.6 *Een **byte** is een rij van 8 bits. Een byte wordt afgekort als B (hoofdletter B).*

Definitie 1.7 ***msb** staat voor most significant bit of dus de meeste linkse bit in een binair getal (zie figuur 1.9).*

Definitie 1.8 ***lsb** staat voor least significant bit of dus de meeste rechtse bit in een binair getal (zie figuur 1.9).*



Figuur 1.9: msb en lsb

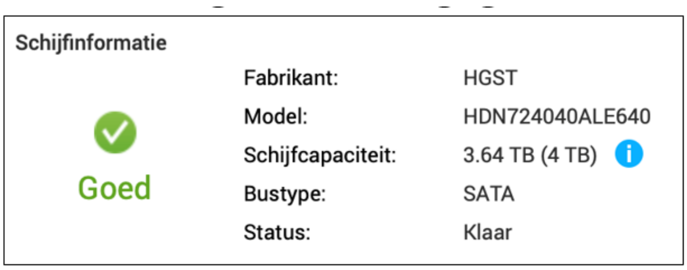
1.2.2.6 Veelvouden van bytes

In figuur 1.10 worden de meest gebruikte veelvouden van zowel decimale getallen (kolom onder SI-voorvoegsels) als van binaire getallen weergegeven (kolom onder Binaire voorvoegsels). In de derde kolom worden de procentuele afwijkingen weergegeven, tussen de voorvoegsels in SI en binair. Merk op dat naarmate men hoger gaat, de afwijking significanter wordt.

Dit brengt ons bij de volgende veel voorkomende praktijk. In software wordt soms de foutieve eenheid gebruikt om bijvoorbeeld de opslagruimte van een harde schijf te noteren (zie figuur 1.11).

SI-voorvoegsels		Binaire voorvoegsels		
Symbool (naam)	Waarde	Symbool (naam)	Waarde	Afwijking tussen SI en binair
kB (kilobyte)	$1000^1 = 10^3$	KiB (kibibyte)	$1024^1 = 2^{10}$	2,4%
MB (megabyte)	$1000^2 = 10^6$	MiB (mebibyte)	$1024^2 = 2^{20}$	4,9%
GB (gigabyte)	$1000^3 = 10^9$	GiB (gibibyte)	$1024^3 = 2^{30}$	7,4%
TB (terabyte)	$1000^4 = 10^{12}$	TiB (tebibyte)	$1024^4 = 2^{40}$	10,0%
PB (petabyte)	$1000^5 = 10^{15}$	PiB (pebibyte)	$1024^5 = 2^{50}$	12,6%
EB (exabyte)	$1000^6 = 10^{18}$	EiB (exbibyte)	$1024^6 = 2^{60}$	15,3%
ZB (zettabyte)	$1000^7 = 10^{21}$	ZiB (zebibyte)	$1024^7 = 2^{70}$	18,1%
YB (yottabyte)	$1000^8 = 10^{24}$	YiB (yobibyte)	$1024^8 = 2^{80}$	20,9%

Figuur 1.10: SI- en binaire voorvoegsels



Figuur 1.11: Specificaties van een harde schijf

De opslagruimte van een harde schijf van 4TB wordt hier genoteerd als 3,64TB. Dit wordt via het info-knopje (zie figuur 1.12) wel uitgelegd maar eigenlijk is de eenheid foutief. Het zou 3,64TiB moeten zijn! Het verschil is, zoals in figuur 1.10 aangegeven, ongeveer 10%!

> De waarde tussen haakjes is de schijfcapaciteit in decimale notatie (1GB = 1.000.000.000 bytes). De meeste schijffabrikanten gebruiken de decimale notatie. Het NAS toont de opslagruimte in binaire notatie (1GB = 1.073.741.824 bytes), die lager kan lijken maar dezelfde hoeveelheid ruimte is.

Figuur 1.12: Uitleg via info-knopje

1.2.3 Octale getallen

1.2.3.1 Voorstelling

Het octaal talstelsel of de octale getallen vormen een positioneel talstelsel met als grondtal, het getal 8 en als verzameling cijfers 0, 1, 2, 3, 4, 5, 6, 7. Voorbeeld van een octaal getal wordt weergegeven in figuur 1.13. De betekenis van elk symbool wordt weergegeven in figuur 1.14.

$$(172)_8$$

Figuur 1.13: Voorstelling octaal getal

8^2	8^1	8^0
1	7	2

Figuur 1.14: Betekenis symbolen in octale getallen

Opnieuw worden de machten van het grondtal, nu het getal 8, neergeschreven, opgaand van rechts naar links, met daaronder de cijfers van het getal op de overeenstemmende posities.

Merk op: *om onderscheid te maken met de tiendelige getallen worden octale getallen in IT-omgevingen vooraf gegaan door de prefix 0 (nul).* Toepassing zie figuur 1.15

$$(172)_8 = 0172$$

Figuur 1.15: Octale voorstelling met subscript en met prefix 0

1.2.3.2 Octaal tellen

Octaal tellen gebeurt analoog als bij decimaal en binair tellen, zoals weergegeven in figuur 1.16.

In figuur 1.16 zijn eveneens de overeenkomstige binaire en decimale waarden weergegeven.

Octale waarden	Binaire waarden	Decimale waarden
0	000	0
1	001	1
2	010	2
3	011	3
4	100	4
5	101	5
6	110	6
7	111	7
10	001 000	8
11	001 001	9
12	001 010	10
13	001 011	11
14	001 100	12
15	001 101	13

Figuur 1.16: Octaal, binair en decimaal tellen

Merk op dat we elk octaal symbool kunnen voorstellen door maximaal **drie bits**. Aangezien voorloopnullen (de groene nullen in de figuur) geen waarde toevoegen aan een getal, in om het even welk talstelsel, kan zelfs elk octaal cijfer weergegeven worden door exact drie bits.

Zoals we hierboven zagen kan elk octaal getal voorgesteld worden door exact 3 bits.Dit heeft natuurlijk te maken met het feit dat 8 gelijk is aan de derde macht van 2 (zie figuur 1.17)

$$2^3 = 8$$

Figuur 1.17: Octale cijfers bestaan uit 3 bits

We kunnen hiervan nu handig gebruik maken, om een binair getal om te zetten naar zijn octale voorstelling. Neem het voorbeeld uit figuur 1.18.

$$(111010)_2 = (\ 111\ 010\ )_2 = (72)_8$$

Figuur 1.18: Voorbeeld omzetting binair naar octaal

We groeperen (beginnend vanaf rechts) de bits in het binair getal in groepjes van 3 bits. Per 3 bits bepalen we de octale waarde via figuur 1.16. Indien het laatste, dus meest linkse groepje, niet exact 3 bits heeft voegen we eventueel voorloopnullen toe.

1.2.3.3 Toepassingsvoorbeeld

Als toepassingsvoorbeeld beschouwen we de bestandspermissies in Linux. Dit is één van de meest frequente voorbeelden waarbij een IT-er in contact komt met octale cijfers.

Algemeen bestaat de bestandenpermissies uit 3 groepen permissies: de permissies voor de eigenaar of owner, permissies van de groepseigenaar of group en de permissies voor iedereen of all (zie figuur 1.19).

- $(rwx)_{owner}$ $(rwx)_{group}$ $(rwx)_{all}$ *bestandsNaam*

Figuur 1.19: Bestandenpermissies in Linux

Zoals in figuur 1.19 te zien is, zijn voor elke groep permissies drie bits voorzien: r-bit voor 'read' of leesrecht, w-bit voor 'write' of schrijfrecht en x-bit voor 'execute' of recht van uitvoeren.
Afhankelijk van wat toegelaten is (= 1) of niet toegelaten is (= 0) kunnen de bestandsrechten weergegeven worden in bit-vlaggen. Een voorbeeld hiervan is te zien in figuur 1.20.

- $(111)_{owner}$ $(101)_{group}$ $(100)_{all}$ *bestandsNaam*

Figuur 1.20: Voorbeeld binaire voorstelling permissies in Linux

Figuur 1.21 geeft dan uiteindelijk de octale weergave van de bestandspermissies.

- $(7)_{owner}$ $(5)_{group}$ $(4)_{all}$ *bestandsNaam*

Of

- 754 *bestandsNaam*

Figuur 1.21: Voorbeeld octale voorstelling permissies in Linux

1.2.4 Hexadecimale getallen

1.2.4.1 Voorstelling

De hexadecimale getallen vormen een positioneel talstelsel met als grondtal, het getal 16 en als verzameling cijfers = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A (=10), B (=11), C (=12), D (=13), E (=14), F (=15). Voorbeeld van een hexadecimaal getal wordt weergegeven in figuur 1.22. De betekenis van elk symbool wordt weergegeven in figuur 1.23.

$$(1272)_{16}$$

Figuur 1.22: Voorstelling hexadecimaal getal

16^3	16^2	16^1	16^0
1	2	7	2

Figuur 1.23: Betekenis symbolen in hexadecimale getallen

Opnieuw worden de machten van het grondtal, 16, neergeschreven, opgaand van rechts naar links, met daaronder de cijfers van het getal op de overeenstemmende posities. Merk op: om onderscheid te maken met de tiendelige getallen worden hexadecimale getallen in IT-omgevingen vooraf gegaan door de prefix 0x (nulx). Toepassing zie figuur 1.24.

$$(1272)_{16} = 0x1272$$

Figuur 1.24: Hexadecimale voorstelling met subscript en met prefix 0x

1.2.4.2 Hexadecimaal tellen

Hexadecimaal tellen gebeurt analoog als bij decimaal, binair en octaal tellen, zoals weergegeven in figuur 1.25.

Hexadecimale waarden	Binaire waarden	Decimale waarden
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
A	1010	10
B	1011	11
C	1100	12
D	1101	13
E	1110	14
F	1111	15
10	0001 0000	16
11	0001 0001	17

Figuur 1.25: Hexadecimaal tellen

In figuur 1.25 zijn eveneens de overeenkomstige binaire en decimale waarden weergegeven. Merk op dat we elk hexadecimaal symbool kunnen voorstellen door vier bits, eventueel gebruik makend van voorloopnullen (de groene nullen in de figuur), omdat 16 gelijk is aan de vierde macht van 2 (zie figuur 1.26)

$$2^4 = 16$$

Figuur 1.26: 1 hexadecimaal getal bestaat uit 4 bits

Om een binair getal om te zetten naar zijn hexadecimale voorstelling, groeperen we nu in groepjes van vier bits. Neem het voorbeeld uit figuur 1.27.

$$(111010)_2 = (0011\ 1010)_2 = (3A)_{16}$$

Figuur 1.27: Omzetting binair naar hexadecimaal

We groeperen (beginnend vanaf rechts) de bits in het binair getal in groepjes van 4 bits. Per 4 bits bepalen we de hexadecimale waarde via figuur 1.25. Indien het laatste, dus meest linkse groepje, niet exact 4 bits heeft voegen we eventueel voorloopnullen (groen in figuur 1.27) toe.

1.2.4.3 Toepassingsvoorbeelden

- **Voorbeeld 1: IPv6-adres**:
  Het IPv6-adres is de opvolger van het IPv4-adres en wordt gebruikt in dezelfde omstandigheden als deze laatste, alleen met het grote verschil dat het met IPv6 mogelijk is veel meer adressen te voorzien. Een IPv6-adres bestaat uit 128 bits wat dus veel groter is dan de 32 bits van het IPv4-adres.
  Deze 128 bits worden weergegeven door 32 hexadecimale getallen, verdeeld in groepjes van 4 hexadecimale getallen, gescheiden door dubbelpunt.

  Bijvoorbeeld: 2001:0DB8:ACAD:0001:0000:0000:0000:0002

- **Voorbeeld 2: MAC-adres**:
  Het MAC-adres van een netwerkkaart is een fabrikantafhankelijk fysiek adres van deze kaart. Zulk een MAC-adres bestaat uit 48 bits die weergegeven worden door 12 hexadecimale getallen.
  De voorstellingswijzen van MAC-adressen, kunnen uiteenlopend zijn: de hexadecimale getallen kunnen ofwel per 2 of per 4 gegroepeerd zijn.

  Voorbeelden: 00:0C:6E:C1:22:4A of 000C – 6EC1 – 224A

1.3 Conversies tussen talstelsels

1.3.1 Binaire combinaties

Zoals reeds eerder uitgelegd, is het binaire stelsel een positioneel stelsel met als grondtal 2 en bevat de symbolenverzameling, 2 symbolen namelijk 0 en 1.

Dit houdt dus in dat een bit of een getal bestaande uit één bit, twee waarden kan aannemen of dus 2 verschillende mogelijkheden heeft, namelijk 0 of 1.

Indien een binair getal uit twee bits bestaat kunnen we hiermee vier verschillende combinaties maken namelijk 00, 01, 10, 11.

In het geval het binair getal uit 3 bit bestaat, kan dit getal 8 verschillende waarden aannemen, enz.

Algemeen kan men bepalen, wanneer we het aantal bit definiëren als n en het grondtal 2 is, kunnen we het aantal mogelijke combinaties die we als x voorstellen, bepalen door de formule in figuur 1.28:

$$x=2^n$$

Figuur 1.28: Aantal combinaties voor n bits

Deze formule biedt antwoord op vragen als : "Hoeveel bit is er nodig om x verschillende getallen voor te stellen?" of "Hoeveel adreslijnen zijn er nodig om x geheugenplaatsen te adresseren?"

Omdat machten van 2 frequent voorkomen en gebruikt worden in de informatica, wordt er verwacht om de elementaire machten van 2 van buiten te kennen (zie figuur 1.29).

$2^0 = \mathbf{1}$; $2^1 = \mathbf{2}$; $2^2 = \mathbf{4}$; $2^3 = \mathbf{8}$; $2^4 = \mathbf{16}$; $2^5 = \mathbf{32}$; $2^6 = \mathbf{64}$;
$2^7 = \mathbf{128}$; $2^8 = \mathbf{256}$; $2^9 = \mathbf{512}$; $\mathbf{2^{10} = 1024 = 1\ kilo}$; ...

Figuur 1.29: Van buiten te kennen machten van 2

De rekenregels uit figuur 1.30, kunnen bij berekeningen handig zijn.

$$a^{n+m} = a^n.a^m \quad \text{en} \quad a^n/a^m = a^{n-m}$$

Figuur 1.30: Rekenregels machten

- **Voorbeeld 1**:Hoeveel mogelijke combinaties kunnen we vormen uit een getal met 9 bits?

  Antwoord:

  $2^9 = 512$ mogelijke combinaties

- **Voorbeeld 2**:Uit hoeveel bit moet een binair getal bestaan om 2048 mogelijke combinaties te kunnen vormen?

  Antwoord:

  $2048 = 2^{11}$ => het binaire getal moet uit 11 bits bestaan

1.3.2 Conversie decimaal binair

Bij conversies, delen we de cijfers van het, om te zetten, getal in, in het deel voor de komma en het deel na de komma.

1.3.2.1 Cijfers voor de komma (methode 1)

Bij de conversie van decimaal naar binair of omgekeerd is het, zoals reeds vermeld, van belang dat de machten van 2 gekend zijn. Deze machten worden nog eens weergegeven in figuur 1.31.

2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
1024	512	256	128	64	32	16	8	4	2	1

Figuur 1.31: Van buiten te kennen machten van 2

In deze figuur zijn de machten van 2 gerangschikt van rechts naar links volgens oplopende machten.

Dit schema zal gebruikt worden bij de conversies decimaal naar binair of omgekeerd.

De conversies op basis van de ***machten van 2***, wordt in het vervolg van deze cursus aangeduid als ***methode 1***.

1.3.2.1.1 Omzetten van binair naar decimaal

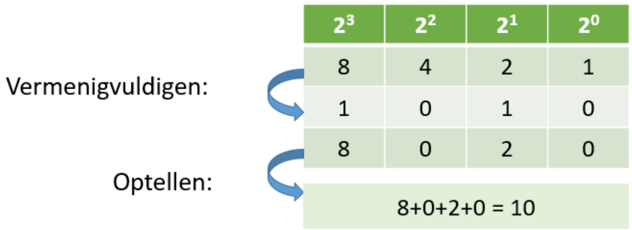
Het omzetten van binair naar decimaal wordt toegelicht aan de hand van het voorbeeld in figuur 1.32 en figuur 1.33.

Vertrek van de lsb van het binair getal en voeg de bits toe in de tabel, één voor één, van rechts naar links.

$$(1010)_2$$

Figuur 1.32: Om te zetten binair getal

Vermenigvuldig dan elke bit met de waarde van de macht van 2, dat net boven de bit staat. Tel dan uiteindelijk alle bekomen resultaten op, zoals in figuur 1.33.



Figuur 1.33: Binair getal ingevuld in tabel met machten van 2

Naarmate men hier vlotter in wordt, kan men de machten van 2 onmiddellijk uitschrijven in een som. Voorbeeld zie figuur 1.34

$$\begin{aligned}(1010)_2 &= 1.2^3 + 0.2^2 + 1.2^1 + 0.2^0 \\ &= 1.8 \quad + 1.2 \quad = (10)_{10}\end{aligned}$$

Figuur 1.34: Onmiddellijk uitschrijven van de machten van 2

1.3.2.1.2 Omzetten van decimaal naar binair

Laten we nog eens de nadruk leggen op het belang van het kennen van de machten van 2. Om een decimaal getal om te vormen naar zijn binaire waarde is het essentieel om de machten van 2 te herkennen.

Algemeen kan de omzetting als volgt beschreven worden:

- Voor een gegeven decimaal getal, zoek naar de hoogste macht van 2 kleiner dan het getal. Trek deze waarde van het oorspronkelijk decimale getal af.
- Zoek opnieuw naar de hoogste macht van 2 kleiner dan het bekomen verschil en bepaal opnieuw het verschil.
- Je doet zo verder tot wanneer je 0 uitkomt.
- Noteer de gebruikte machten van 2.

Maak een tabel van opeenvolgende machten van 2, vertrekkende van de hoogste gebruikte macht, analoog aan figuur 1.31, en schrijf een 1 onder de gebruikte machten en een 0 onder de niet gebruikte machten

De onderste rij van 0-len en 1-en vormen de binaire waarde van het oorspronkelijke decimale getal.

Figuur 1.35 geeft, ter illustratie, de omzetting weer van 862 naar zijn binaire waarde.

Voorbeeld: $(862)_{10}$

- Stap 1: zoek naar de hoogste macht van 2 kleiner dan het getal:
  $2^9 = 512$
- Stap 2: trek deze waarde van het getal af :
  862 – 512 = **350**
- Herhaal stap 1 en 2 op het resultaat van stap 2 tot wanneer je 0 uitkomt:
  2^8 = 256 -> 350 -256 = **94** → 2^6 = 64 -> 94 - 64 = **30** → 2^4 = 16 -> 30 - 16 = **14** →
  2^3 = 8 -> 14 - 8 = 6 → 2^2 = 4 -> 6 – 4 = 2 → 2^1 = 2 -> 2 – 2 = 0
- Maak een tabel van opeenvolgende machten van 2, vertrekkende van de hoogste gebruikte macht, en schrijf een 1 onder de gebruikte machten en een 0 onder de niet gebruikte machten:

2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
1	1	0	1	0	1	1	1	1	0

- Antwoord: $(862)_{10} = (1101011110)_2$

Figuur 1.35: Omzetting van de decimale waarde 862 naar binair

1.3.2.2 Cijfers na de komma (methode 1)

Ook hier is het weer essentieel, om de machten van 2 te kennen.

Voor cijfers na de komma maken we gebruik van de negatieve machten van 2. We zullen deze noteren naast elkaar van links naar rechts, volgens dalende machten, zoals weergegeven in figuur 1.36.

2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}
0,5	0,25	0,125	0,0625	0,03125

Figuur 1.36: Tabel met negatieve machten van 2

1.3.2.2.1 Omzetten van binair naar decimaal

De omzetting wordt toegelicht in figuur 1.37 en 1.38.

We vertrekken vanaf het eerste cijfer na de komma, en voegen dit toe in de tabel van figuur 1.38, van links naar rechts.

$$(0{,}011)_2$$

Figuur 1.37: Om te zetten binair getal

	2^{-1}	2^{-2}	2^{-3}
	0,5	0,25	0,125
Vermenigvuldigen:	0	1	1
	0	0,25	0,125
Optellen:	0+0,25+ 0,125 = 0,375		

Figuur 1.38: Tabel voor omzetting binair na komma

Vermenigvuldig dan elke bit met de waarde van de macht van 2, dat net boven de bit staat. Tel dan uiteindelijk alle bekomen resultaten op.

Naarmate men hier vlotter in wordt, kan men de machten van 2 onmiddellijk uitschrijven in een som. Voorbeeld zie figuur 1.39

$$\begin{aligned}(0{,}011)_2 &= 0.2^{-1} + 1.2^{-2} + 1.2^{-3} \\ &= 1\ .\ 0{,}25 + 1\ .\ 0{,}125 = (0{,}375)_{10}\end{aligned}$$

Figuur 1.39: Omzetting door uitschrijving van negatieve machten van 2

1.3.2.2.2 Omzetten van decimaal naar binair

We werken opnieuw volgens het principe van het herkennen van, nu de negatieve, machten van 2.

Algemeen kan de omzetting als volgt beschreven worden:

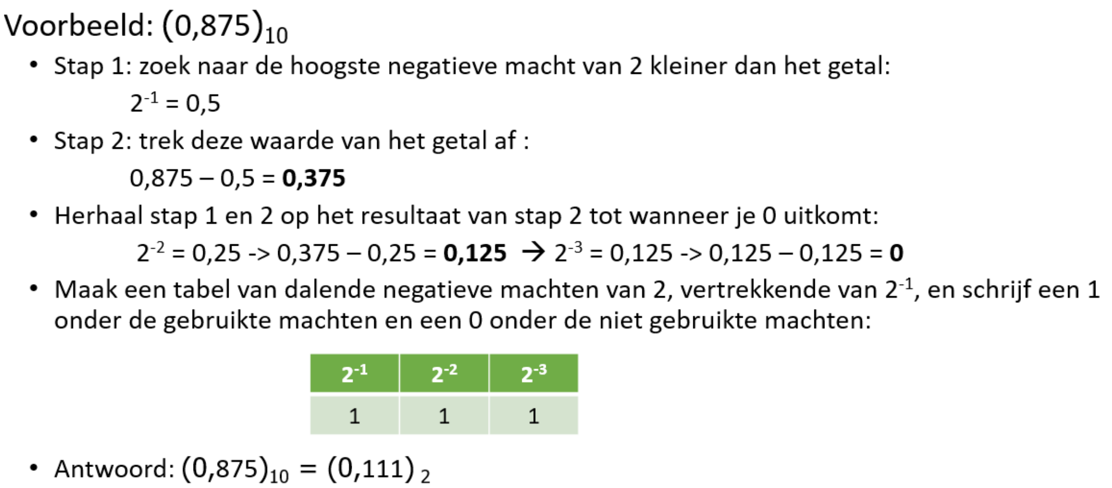
- Zoek naar de hoogste negatieve macht van 2 kleiner dan het gegeven decimaal getal.
- Trek deze waarde van het oorspronkelijk decimale getal af.

- Herhaal deze stappen.
- Je doet zo verder tot wanneer je 0 uitkomt.
- Noteer de gebruikte machten van 2.

Maak een tabel negatieve machten van 2, analoog aan figuur 1.36, en schrijf een 1 onder de gebruikte machten en een 0 onder de niet gebruikte machten.

De onderste rij van 0-len en 1-en vormen getallen na de komma van de binaire waarde van het oorspronkelijke decimale getal.

Figuur 1.40 geeft, ter illustratie, de omzetting weer van 0,875 naar zijn binaire waarde.


Voorbeeld: $(0{,}875)_{10}$

- Stap 1: zoek naar de hoogste negatieve macht van 2 kleiner dan het getal:
  $2^{-1} = 0{,}5$
- Stap 2: trek deze waarde van het getal af :
  $0{,}875 - 0{,}5 = \mathbf{0{,}375}$
- Herhaal stap 1 en 2 op het resultaat van stap 2 tot wanneer je 0 uitkomt:
  $2^{-2} = 0{,}25$ -> $0{,}375 - 0{,}25 = \mathbf{0{,}125}$ → $2^{-3} = 0{,}125$ -> $0{,}125 - 0{,}125 = \mathbf{0}$
- Maak een tabel van dalende negatieve machten van 2, vertrekkende van 2^{-1}, en schrijf een 1 onder de gebruikte machten en een 0 onder de niet gebruikte machten:

2^{-1}	2^{-2}	2^{-3}
1	1	1

- Antwoord: $(0{,}875)_{10} = (0{,}111)_2$

Figuur 1.40: Omzetting van de decimale waarde 0,875 naar binair

1.3.2.3 Cijfers na de komma (methode 2)

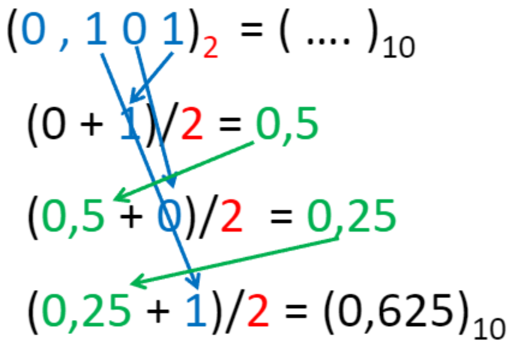
Het uitschrijven of herkennen van machten is enkel praktisch voor omzettingen van binair naar decimaal of omgekeerd, indien het getal een beperkt aantal cijfers na de komma heeft.

Wat nu in het geval dat er na de komma teveel cijfers zijn en/of je geen herkenning van een macht kunt toepassen, omdat het geen duidelijke sommen zijn machten van 2?

In beide gevallen, kan je gebruik maken volgende methodes: voor de omzetting van binair naar decimaal, gebruik dan de methode van de opeenvolgende delingen; voor de omzetting van decimaal naar binair, gebruik de methode van de opeenvolgende vermenigvuldigingen

1.3.2.3.1 Omzetten van binair naar decimaal of methode van de opeenvolgende delingen

We illustreren de methode van de opeenvolgende delingen aan de hand van een eenvoudig voorbeeld: de omzetting van het binair getal 0,101 naar zijn decimale waarde (zie figuur 1.41).



Figuur 1.41: Omzetting van binair getal 0,101 naar decimale waarde

Figuur 1.41 in detail:

We beginnen met 0. Dan tellen we er de meest rechtse bit, in dit geval de rechtse 1, bij op (blauwe pijl), en delen het resultaat door 2. In totaal is dit 0,5 (in het groen). Met dit resultaat herhalen we de bewerking: we tellen er de volgende bit, een 0, bij op, en delen het geheel door 2. We bekomen dan 0,25. We herhalen deze stappen voor de laatste (meest linkse) bit na de komma: optellen bij 0.25, en delen door 2, voor een eindtotaal van 0.625. Hier stopt de iteratie en we lezen het eindresultaat of dus de decimale waarde af van het binaire getal, 0,101.

1.3.2.3.2 Omzetten van decimaal naar binair of methode van de opeenvolgende vermenigvuldigingen

De omzetting van decimaal naar binair met de methode van de opeenvolgende vermenigvuldigingen wordt gedemonstreerd aan de hand van de omzetting van 0,375 naar zijn binaire waarde in figuur 1.42.

Figuur 1.42 in detail:

We vertrekken nu vanuit het volledige decimale getal 0,375. We vermenigvuldigen deze waarde met 2. Vervolgens houden we het bekomen cijfer voor de komma opzij. De volgende lijn doen we enkel verder met het deel na de komma van de voorgaande lijn. Vervolgens worden voorgaande stappen herhaald. Er wordt gestopt wanneer er geen cijfers meer na de komma zijn als gevolg van de vermenigvuldiging met 2. Het eindresultaat zijn de cijfers na de komma, te lezen van boven naar onder, voor de binaire waarde van 0,375.

$(0{,}375)_{10} = (\ldots)_2$

$0{,}375 \,.\, 2 = 0{,}75 \rightarrow 0$ (cijfer voor de komma)

$0{,}75 \,.\, 2 = 1{,}5 \rightarrow 1$

$0{,}5 \,.\, 2 = 1 \rightarrow 1$

$(0{,}375)_{10} = (0{,}011)_2$

Figuur 1.42: Omzetting van 0,375 naar binaire waarde

1.3.3 Conversies talstelsels met als basis een macht van 2

In geval dat de basis van een talstelsel een macht van 2 is, zoals het octaal en het hexadecimaal talstelsel, zal één symbool van dat talstelsel voorgesteld kunnen worden door een vast aantal bits. Zo wordt elk symbool van het octaal talstelsel voorgesteld door 3 bits, omdat de basis van dit talstelsel 8 is of dus de derde macht van 2. In het hexadecimaal stelsel wordt elk symbool voorgesteld door 4 bits omdat in dit geval de basis 16 of dus twee tot de vierde macht is.

Hiervan maken we handig gebruik voor de omzettingen. Voor het omzetten van het talstelsel naar de binaire getallen, stel elk symbool voor door de overeenkomstige groep bits (zie figuur 1.43).

Voor de omzetting van binair naar het talstelsel: groepeer in het binaire getal de symbolen, startend bij de komma en zet de groepen om naar het overeenkomstig cijfer in het talstelsel volgens figuur 1.43.

Om volledige groepen te vormen voeg dan eventueel voor- en achteraan nullen toevoegen.

In figuur 1.44 worden een aantal voorbeelden toegelicht van omzettingen.

Octale waarden	Binaire waarden	Decimale waarden
0	000	0
1	001	1
2	010	2
3	011	3
4	100	4
5	101	5
6	110	6
7	111	7
10	001 000	8
11	001 001	9
12	001 010	10
13	001 011	11
14	001 100	12
15	001 101	13

Hexadecimale waarden	Binaire waarden	Decimale waarden
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
A	1010	10
B	1011	11
C	1100	12
D	1101	13
E	1110	14
F	1111	15
10	0001 0000	16
11	0001 0001	17

Figuur 1.43: Tabellen voor de omzettingen octaal en hexadecimaal

Voorbeelden:

$(11010{,}0100001)_2 = (0001\ 1010{,}0100\ 0010)_2 = (1A{,}42)_{16}$

$(11010{,}0100001)_2 = (011\ 010{,}010\ 000\ 100)_2 = (32{,}204)_8$

$(7{,}A8)_{16} = (0111{,}1010\ 1000)_2 = (111{,}10101)_2$

$(4{,}17)_{16} = (0100{,}0001\ 0111)_2 = (100{,}00010111)_2$

$(4{,}17)_8 = (100{,}001\ 111)_2 = (100{,}001111)_2$

Figuur 1.44: Voorbeelden voor omzettingen tussen verschillende talstelsels

1.3.4 Oefeningen op conversies

- **Oefening 1:**

Reken om

1. $(100110010011)_2 = (\ldots\ldots\ldots\ldots\ldots)_{16} = (\ldots\ldots\ldots\ldots\ldots)_8$
2. $(1001110011110010)_2 = (\ldots\ldots\ldots\ldots\ldots)_{16} = (\ldots\ldots\ldots\ldots\ldots)_8$
3. $(101010111100)_2 = (\ldots\ldots\ldots\ldots\ldots)_{16} = (\ldots\ldots\ldots\ldots\ldots)_8$
4. $(1FD)_{16} = (\ldots\ldots\ldots\ldots\ldots)_2$
5. $(CCC)_{16} = (\ldots\ldots\ldots\ldots\ldots)_2$
6. $(5307)_8 = (\ldots\ldots\ldots\ldots\ldots)_2$
7. $(7264)_8 = (\ldots\ldots\ldots\ldots\ldots)_2$
8. $(2A5C)_{16} = (\ldots\ldots\ldots\ldots\ldots)_8$
9. $(243)_8 = (\ldots\ldots\ldots\ldots\ldots)_{16}$

- **Oefening 2:**

Reken om

1. $(100110{,}010011)_2 = (\ldots\ldots\ldots\ldots\ldots)_{16} = (\ldots\ldots\ldots\ldots\ldots)_8$
2. $(1011100{,}1111001)_2 = (\ldots\ldots\ldots\ldots\ldots)_{16} = (\ldots\ldots\ldots\ldots\ldots)_8$
3. $(10101{,}01111)_2 = (\ldots\ldots\ldots\ldots\ldots)_{16} = (\ldots\ldots\ldots\ldots\ldots)_8$
4. $(1F{,}D)_{16} = (\ldots\ldots\ldots\ldots\ldots)_2$
5. $(C{,}CC)_{16} = (\ldots\ldots\ldots\ldots\ldots)_2$
6. $(53{,}07)_8 = (\ldots\ldots\ldots\ldots\ldots)_2$
7. $(72{,}64)_8 = (\ldots\ldots\ldots\ldots\ldots)_2$
8. $(2A{,}5C)_{16} = (\ldots\ldots\ldots\ldots\ldots)_8$
9. $(2{,}43)_8 = (\ldots\ldots\ldots\ldots\ldots)_{16}$

- **Oefening 3:**

Reken om

1. $(10000100)_2 = (\ldots\ldots\ldots\ldots\ldots)_{10}$
2. $(1000010001)_2 = (\ldots\ldots\ldots\ldots\ldots)_{10}$
3. $(100100001)_2 = (\ldots\ldots\ldots\ldots\ldots)_{10}$
4. $(131)_{10} = (\ldots\ldots\ldots\ldots\ldots)_2$
5. $(260)_{10} = (\ldots\ldots\ldots\ldots\ldots)_2$
6. $(15)_{10} = (\ldots\ldots\ldots\ldots\ldots)_2$

- **Oefening 4:**

Reken om

1. $(1000{,}0110)_2 = (\ldots\ldots\ldots\ldots\ldots)_{10}$
2. $(111001)_2 = (\ldots\ldots\ldots\ldots\ldots)_{10}$
3. $(100010010{,}0011)_2 = (\ldots\ldots\ldots\ldots\ldots)_{10}$
4. $(10101100{,}011)_2 = (\ldots\ldots\ldots\ldots\ldots)_{10}$
5. $(0{,}1111)_2 = (\ldots\ldots\ldots\ldots\ldots)_{10}$
6. $(12{,}15)_8 = (\ldots\ldots\ldots\ldots\ldots)_{10}$
7. $(26{,}7)_8 = (\ldots\ldots\ldots\ldots\ldots)_{10}$
8. $(1A{,}4)_{16} = (\ldots\ldots\ldots\ldots\ldots)_{10}$
9. $(1A{,}18)_{16} = (\ldots\ldots\ldots\ldots\ldots)_{10}$

- **Oefening 5:**

Reken om

1. $(11{,}5625)_{10}$ = $(\ldots\ldots\ldots\ldots\ldots)_2$
2. $(36)_{10}$ = $(\ldots\ldots\ldots\ldots\ldots)_2$
3. $(172)_{10}$ = $(\ldots\ldots\ldots\ldots\ldots)_2$
4. $(17{,}375)_{10}$ = $(\ldots\ldots\ldots\ldots\ldots)_2$
5. $(89{,}625)_{10}$ = $(\ldots\ldots\ldots\ldots\ldots)_8$
6. $(126{,}25)_{10}$ = $(\ldots\ldots\ldots\ldots\ldots)_{16}$
7. $(86{,}2)_{10}$ = $(\ldots\ldots\ldots\ldots\ldots)_{16}$
8. $(67{,}3)_{10}$ = $(\ldots\ldots\ldots\ldots\ldots)_{16}$
9. $(1984)_{10}$ = $(\ldots\ldots\ldots\ldots\ldots)_{16}$

- **Oefening 6: oefening op IPv4-adressen**

  Conversies tussen het decimale, binaire en hexadecimale talstelsels worden voortdurend toegepast bij het berekenen van zowel IPv4- als IPv6-adressen. Even een introductie op IPv4 adressen. Deze adressen worden gebruikt om te communiceren tussen verschillende toestellen op een netwerk.

  Een IPv4-adres bestaat uit 32 bits opgedeeld in 2 delen namelijk een **netwerkwerkgedeelte** en een **hostgedeelte**. Het **netwerkgedeelte** vertelt je tot welk netwerk je apparaat behoort en zijn de bits, aaneensluitend aan de linkerkant van het adres. Het **hostgedeelte** vertelt je welk apparaat, je toestel binnen een netwerk is, en zijn de bits aaneensluitend aan de rechterkant.

  De scheiding tussen netwerkbits en hostbits wordt weergegeven ofwel door het subnetmask ofwel door de prefixnotatie. Hierover leer je verder in het opleidingsonderdeel Computernetworks I.

  Vervolgens, beschikt elk netwerk over een netwerkadres, waarbij alle bits van het hostgedeelte op 0 staan.

  Eveneens beschikt een netwerk over een broadcastadres, waarbij alle bits van het hostgedeelte op 1 staan.

  Daarnaast, heeft , elk netwerk ook een adresrange waaruit een IP-adres kan gekozen worden om toe te kennen aan een apparaat binnen een bepaald netwerk. De adresrange is het interval tussen het netwerkadres en het broadcastadres van een netwerk.

  Een IPv4-adres heeft twee voorstellingen namelijk, een binaire voorstelling, bestaande uit 32 bits, en een dotted decimal voorstelling, bestaande uit 4 decimale cijfers met een punt tussen.

  Om over te stappen van de binaire voorstelling naar de dotted decimal voorstelling groeperen we de bitreeks per acht bit, of dus per Byte, waarbij van elk groepje de decimale waarde bepaald wordt. Zo bekomen we bijvoorbeeld 192.168.16.5, als IPv4-adres.

Opgave:

Gegeven: Een IPv4-adres 192.168.78.64

Gevraagd: Bereken de hexadecimale waarde van dit IP-adres.

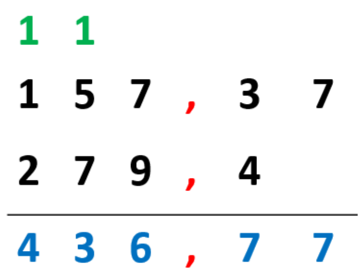
1.4 Bewerkingen in het binair stelsel

1.4.1 Optellen in het binair stelsel

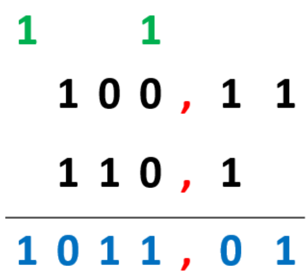
De binaire optelling kan op compleet dezelfde manier uitgevoerd worden zoals in het decimaal stelsel. De werkwijze bestaat uit drie opeenvolgende stappen:

1. Schrijf de twee getallen onder elkaar met de komma's onder elkaar
2. Tel cijfer per cijfer bij elkaar op, van rechts naar links
3. Draag indien nodig een één (1) over naar een hogere rangorde

In figuur 1.45 wordt deze werkwijze getoond voor de optelling van twee decimale getallen en in figuur 1.46 de optelling van twee binaire getallen.



Figuur 1.45: Optelling van decimale getallen



Figuur 1.46: Optelling van binaire getallen

1.4.2 Oefeningen op optellen in het binair stelsel

Maak onderstaande optellingen in het binair stelsel. Converteer waar nodig eerst van decimaal naar binair om, vervolgens, de binaire optelling uit te voeren. Als controle kan je het resultaat opnieuw converteren naar het decimale talstelsel om te zien of je de optelling correct hebt uitgevoerd.

1. $(1011)_2 + (10101)_2 = (\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots)_2$
2. $(110{,}11)_2 + (10{,}101)_2 = (\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots)_2$
3. $(23{,}25)_{10} + (40{,}5)_{10} = (\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots)_2$
4. $(100)_{10} + (28)_{10} = (\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots)_2$
5. $(97)_{10} + (115)_{10} = (\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots)_2$
6. $(147)_{10} + (35)_{10} = (\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots)_2$

1.4.3 Andere bewerkingen

1.4.3.1 Complement(eren)

Complementeren van een binair getal is enkel mogelijk wanneer, er op voorhand afgesproken wordt, hoe groot het getal of computerwoord is, of dus uit hoeveel bits het binair getal bestaat. Bijvoorbeeld: men kan gebruik maken van een 8-bit (of 1 Byte) woord of 16-, 32- of 64-bit woord.

De bewerking complementeren, bestaat eruit de enen naar nullen om te zetten, en de nullen naar enen.

De operator van de bewerking complementeren is het volgend teken: ¬

Een voorbeeld van de complement-bewerking wordt weergegeven in figuur 1.47.

$$\neg[00011010] = [11100101]$$

Figuur 1.47: Complement van een binair getal

1.4.3.2 Bitsgewijze EN

Op twee binaire getallen, ook nog de operanden genoemd, bestaande uit hetzelfde aantal bits, wordt bit per bit de EN-bewerking toegepast. De EN-bewerking, op bit-niveau, gebeurt volgens figuur 1.48. De operator van de EN-bewerking wordt voorgesteld door volgend symbool: Λ

Λ	**0**	**1**
0	0	0
1	0	1

Figuur 1.48: De EN-bewerking

Een voorbeeld van de EN-bewerking wordt weergegeven in figuur 1.49.

[1010] **∧** [1100] = [1000]

Figuur 1.49: Voorbeeld op de EN-bewerking

1.4.3.3 Bitsgewijze OF

In figuur 1.50 wordt de OF-bewerking op bit-niveau getoond.

V	**0**	**1**
0	0	1
1	1	1

Figuur 1.50: De OF-bewerking

Hierbij is de operator van de OF-bewerking, het volgende teken: V Weer wordt op twee operanden, bestaande uit hetzelfde aantal bits, bit per bit de OF-bewerking toegepast zoals weergegeven in figuur 1.51

[10001101] **V** [00111011] = [10111111]

Figuur 1.51: Voorbeeld op de OF-bewerking

1.4.3.4 Bitsgewijze Exclusieve OF of xor

Figuur 1.52 toont de Exclusieve OF of xor op bit-niveau:

$\underline{\mathbf{V}}$	**0**	**1**
0	0	1
1	1	0

Figuur 1.52: De xor-bewerking

De operator wordt als volgt voorgesteld: $\underline{\vee}$ of xor Voor een toepassing van de bewerking xor op twee even lange binaire getallen, zie figuur 1.53.

$$[10001101] \ \underline{\vee} \ [00111011] = [10110110]$$

Figuur 1.53: Voorbeeld op de xor-bewerking

1.4.4 Negatieve getallen

In het decimaal stelsel stellen we een negatief getal voor door een min-teken of '-' te plaatsen voor de absolute waarde van het getal. De bewerking, het verschil of het aftrekken, wordt hierbij gezien als de som van een positief en een negatief getal of, dus, A – B = A + (-B).

Zoals we zullen zien, is het niet zo evident, om dezelfde redenering toe te passen om een negatief binair getal voor te stellen.

Bij het binair stelsel zitten we, immers, met een bijkomend probleem. Omdat binaire getallen per computerwoord (8, 16, 32 of 64 bit) worden bijgehouden in het geheugen en in de CPU, zal de voorstelling van negatieve getallen binnen de beperkte ruimte van het computerwoord moeten gebeuren. Ook het teken van een getal moet binnen het woord bijgehouden worden.

In de zoektocht om een goede voorstelling te vinden, zijn er 3 methodes opgesteld om een negatief getal binair voor te stellen :

1. Teken + absolute waarde
2. Excess-N
3. Two's complement of 2's complement

In het vervolg van de tekst, worden deze 3 methodes besproken, waarbij we ons, voor de eenvoud, beperken tot 8-bit computerwoorden.

1.4.4.1 Teken + absolute waarde

Dit is de analoge methode, als bij de voorstelling van negatieve decimale getallen. We zullen, echter, zien dat dit niet het verhoopte resultaat geeft.

Deze werkwijze bestaat erin door de meest linkse bit, binnen het computerwoord te gebruiken als tekenbit. Hierbij wordt met de nulwaarde, 0, van deze bit, een positief getal aangeduid en met de waarde 1, een negatief getal.

In figuur 1.54 worden de getallen -12 en +22 voorgesteld volgens deze methode

We zien, echter, dat A-B niet kan berekend worden als A+(-B).

-12 → [10001100] en +22 → [00010110]

Figuur 1.54: Binaire voorstelling -12 en +22

Bereken we bijvoorbeeld 22-12 door binair 22 en (-12) bij elkaar op te tellen, dan wordt als eindresultaat niet 10 bekomen maar – 34. Zie figuur 1.55.

```
  00010110
+ 10001100
  10100010      = (-34)10
```

Figuur 1.55: Binaire som van +22 en -12

Dit heeft als gevolg dat de bewerking, het verschil, als afzonderlijke bewerking in de hardware zal moeten geïmplementeerd worden, wat een dure aangelegenheid is! Bewerkingen kunnen, immers, pas uitgevoerd worden nadat beide tekenbits uit de woorden verwijderd zullen zijn en geanalyseerd werden.

Als bijkomend nadeel, heeft het getal 0 twee voorstellingen : 0 en -0 (zie figuur 1.56).

[00000000] en [10000000]

Figuur 1.56: Binaire voorstelling 0 en -0

1.4.4.2. Excess-N

Bij deze methode, wordt bij het om te zetten decimaal getal, een getal N opgeteld.

Hoe groot de waarde van N is, hangt af van de grootte van het bit-woord of, dus, het aantal voorziene bits voor de binaire voorstelling. Zo is voor een 8 bit-woord de waarde voor N = 127, hetgeen overeenkomt met de binaire waarde 0111 1111; voor een 4 bit-woord is N = 7 hetgeen overeenkomt met de binaire waarde 0111 . In figuur 1.57 worden een aantal voorbeelden weergegeven van 8 bit computerwoorden in hun Excess-N voorstelling.

Zoals blijkt uit de figuur wordt nul, nu, slechts op één manier voorgesteld. Dit is al winst opzichte van de voorgaande methode.

Maar het rekenen met negatieve getallen wordt ook hier niet vereenvoudigd:
zo is -127 + (-127) = -127.

Als bijkomend nadeel, wordt, daarnaast, geen enkel getal nog op de gewone binaire manier voorgesteld.

$(1)_{10}$	→ 128	→ [10000000]
$(0)_{10}$	→ 127	→ [01111111]
$(-127)_{10}$	→ 0	→ [00000000]
$(-1)_{10}$	→ 126	→ [01111110]
$(128)_{10}$	→ 255	→ [11111111]

Figuur 1.57: Voorbeelden Excess-N voorstelling van 8 bit-woorden

Een toepassing van deze Excess-N methode, zullen we verder zien bij de floating-point voorstelling.

1.4.4.3. 2's complement

Dit is de meest gebruikte binaire voorstelling voor negatieve getallen omdat deze vorm ook voldoet aan de eigenschap A-B = A+ (-B).

Enkel de negatieve getallen worden op de specifieke 2's complement manier voorgesteld. De positieve getallen blijven in de gewone binaire vorm weergegeven.

Om het 2's complement van een negatief getal te bepalen, worden volgende stappen uitgevoerd:

1. Schrijf de binaire vorm van de absolute waarde
2. Inverteer hiervan alle bits (dus 0 wordt 1 en 1 wordt 0)
3. Tel er binair 1 bij op

Figuur 1.58 toont als voorbeeld de omzetting van decimaal -20 naar zijn 2's complement binaire voorstelling.

1. Schrijf $(20)_{10}$ in binaire vorm → [00010100]
2. Inverteer de bits → [11101011]
3. Tel hierbij 1 binair op:

```
   11
[11101011]
+        1
----------
[11101100]
```

Figuur 1.58: 2's complement binaire voorstelling van -20

In figuur 1.59 worden andere voorbeelden weergegeven.

$(0)_{10}$ → [00000000] $(1)_{10}$ → [00000001]
$(-1)_{10}$ → [11111111] $(-10)_{10}$ → [11110110]

Figuur 1.59: Voorbeelden 2's complement voorstelling

Om nu het onderscheid te maken tussen een positief en een negatief getal, kijk naar de msb of dus de most significant bit, zijnde de meest linkse bit van het computerwoord. Een positief getal heeft bij de 2's complement voorstelling een 0 als msb en een negatief getal heeft een 1 als msb (zie ook figuur 1.59).

Zoals reeds vermeld, geldt bij de 2's complement voorstelling wel dat A-B = A+ (-B). In figuur 1.60 wordt dit aangetoond door middel van het bepalen van het verschil 10 – 20.

$(10)_{10}$ → [00001010]
$(-20)_{10}$ → [11101100]
[11110110]

- **Controle**:
  - [11110110] is een negatief getal, want msb = 1
  - Om de absolute waarde te bepalen:
    - Inverteer eerst de bits: [00001001]
    - Tel er binair 1 bij op: [00001010] = $(10)_{10}$
  - → Dus oplossing = $(-10)_{10}$

Figuur 1.60: Bewerking 10 - 20 in 2's complement voorstelling

1.4.5 Oefeningen op negatieve getallen

Vul onderstaande tabel aan. Gebruik telkens een 8-bit computerwoord.

getal	Teken + abs	Excess-127	2’s-compl.
10			
-13			
43			
-52			
87			
-87			

1.4.6 Overflow

Zoals reeds aangehaald, worden binaire getallen per computerwoord (8, 16, 32 of 64 bit) bijgehouden in het geheugen.

De voorstelling van binaire getallen, zowel positief als negatief zal dus binnen de beperkte ruimte van het computerwoord moeten gebeuren.

Het kan dus gebeuren, dat sommige berekeningen, zoals het optellen van twee 2’s complement getallen, fout lopen omdat het resultaat buiten het bereik van het computerwoord valt.

Nemen we als voorbeeld de som -116 + (-60) = -176. Binair wordt dit (zie figuur 1.61):

$$
\begin{array}{lll}
(-116)_{10} & \rightarrow & [10001100] \\
(-60)_{10} & \rightarrow & \underline{[11000100]} \\
 & & 1[01010000] \rightarrow (80)_{10}
\end{array}
$$

Figuur 1.61: Som -116 + (-60) in 2’s complement voorstelling

In de figuur ziet men duidelijk, dat bij de som de twee tekenbits met waarde 1 bij elkaar worden opgeteld, wat als resultaat een 10 binair geeft, waardoor er een bit nodig is links van de tekenbit of dus buiten het bereik van het computerwoord. Het computerwoord is dus te klein om de binaire waarde van het decimaal getal -176 weer te geven. We spreken van **overflow** omdat het computerwoord als het ware ‘overloopt’.

Een overflow doet zich voor, vanaf het ogenblik een (decimale) getalwaarde zich buiten de range van het computerwoord bevindt.

De range voor een 8 bit computerwoord is decimaal van -127 tot en met 127. Van zodra men een decimaal getal heeft buiten dit interval, als resultaat van een berekening, dan treedt er overflow op.

Binair kan men overflow vaststellen als er zich 1 van de volgende 2 situaties of **overflow condities** zich voordoet:

1. "carry naar het tekenbit" = overdracht van de op één na meest linkse positie naar de meest linkse positie of dus de msb
2. "carry naar buiten" = overdracht van de meest linkse positie (of dus de msb) naar nergens of dus buiten het computerwoord

LET OP: belangrijk is te weten dat er pas overflow zal zijn, als 1 van de twee overflow condities optreedt en NIET beiden tesamen!

Door het optreden van overflow zal , dus, het eindresultaat van de berekening niet correct zijn.

Dit heeft als gevolg, dat er ook 2 situaties zijn, waarbij **geen overflow** zal optreden, namelijk:

1. Er is een carry naar buiten EN een carry naar het tekenbit
2. Geen van beide carries doen zich voor

In het geval geen overflow optreedt, zal het eindresultaat van de bewerking wel correct zijn.

In figuur 1.62 wordt een overzichtstabel gegeven, van de verschillende overflow-situaties:

Carry naar tekenbit	Carry naar buiten	Overflow	Correcte som
Nee	Nee	Nee	Ja
Nee	Ja	Ja	Nee
Ja	Nee	Ja	Nee
Ja	Ja	Nee	Ja

Figuur 1.62: Overzicht overflow-situaties

Hierna volgen vier voorbeelden, van elke overflow-situatie 1 voorbeeld.

In het ***eerste voorbeeld*** wordt de som berekend van 100 + 50. Decimaal geeft dit als resultaat 150. Omdat 150 buiten het bereik van het 8-bit computerwoord ligt, treedt hier overflow op. Welke overflow conditie dit veroorzaakt heeft, zien we in de binaire bewerking in figuur 1.63

		2's complement voorstelling	
		1 1	→ Carry naar tekenbit => overflow
$(100)_{10}$	=	[0 1 1 0 0 1 0 0]	
$(50)_{10}$	=	[0 0 1 1 0 0 1 0]	
		[1 0 0 1 0 1 1 0]	→ $(-106)_{10}$ => verkeerde som

Figuur 1.63: Som 100 + 50 in 2's complement voorstelling

In de figuur is te zien dat de oorzaak van de overflow de carry naar het tekenbit is en dat het resultaat een verkeerde binaire waarde geeft.

Het ***tweede voorbeeld*** toont de som van 100 + (-50). Omdat het decimale resultaat hiervan wel in de range ligt van het 8-bit computerwoord, is er hier geen overflow en is de binaire som correct. In figuur 1.64 wordt de binaire som bepaald. Er is geen overflow want er zijn 2 carries, namelijk, een carry naar tekenbit en een carry naar buiten.

		2's complement voorstelling	
		1 1 1 1	→ Carry naar tekenbit en carry naar buiten => GEEN overflow
$(100)_{10}$	=	[0 1 1 0 0 1 0 0]	
$(-50)_{10}$	=	[1 1 0 0 1 1 1 0]	
		[0 0 1 1 0 0 1 0]	→ $(50)_{10}$ => correcte som

Figuur 1.64: Som 100 + (-50) in 2's complement voorstelling

Het ***derde voorbeeld*** berekend de som van (-100) + (-50). Het decimale resultaat is -150 en ligt buiten het bereik van het 8-bit computerwoord, dus er treedt overflow op. Uit figuur 1.65 blijkt dat een carry naar buiten, de bepalende overflow conditie is, met een verkeerde som als resultaat.

In het ***vierde voorbeeld*** wordt, ten slotte, de som bepaald van (-100) + 50. Het decimale resultaat ligt in de range van het 8-bit computerwoord. Er is geen overflow en er treden ook geen overflow conditities op, zoals te zien is in figuur 1.66. De binaire som is, bijgevolg, ook correct.

	2's complement voorstelling	
	1 1 1 1	→ Carry naar buiten => overflow
$(-100)_{10}$ =	[1 0 0 1 1 1 0 0]	
$(-50)_{10}$ =	[1 1 0 0 1 1 1 0]	
	[0 1 1 0 1 0 1 0]	→ $(106)_{10}$ => verkeerde som

Figuur 1.65: Som (-100) + (-50) in 2's complement voorstelling

	2's complement voorstelling	
	1 1	→ GEEN carry's => GEEN overflow
$(-100)_{10}$ =	[1 0 0 1 1 1 0 0]	
$(50)_{10}$ =	[0 0 1 1 0 0 1 0]	
	[1 1 0 0 1 1 1 0]	→ $(-50)_{10}$ => correcte som

Figuur 1.66: Som (-100) + 50 in 2's complement voorstelling

1.4.7 Oefeningen op overflow

- **Oefening 1:** maak de onderstaande berekeningen in binaire 2's-complement voorstelling in een 8-bit computerwoord en controleer het resultaat.

1. $(-22)_{10} - (12)_{10}$ =
2. $(11)_{10} - (5)_{10}$ =
3. $(-13)_{10} - (15)_{10}$ =
4. $(100)_{10} - (64)_{10}$ =

- **Oefening 2:** maak de onderstaande berekeningen in binaire 2's-complement voorstelling in een 8-bit computerwoord en geef aan of en waarom er een overflow conditie is.

  1. 55 - 25
  2. 100 + 29
  3. -125 + 25
  4. -70 - 80
  5. 43 + 106
  6. 43 – 106
  7. 87 – 52
  8. -20 – 52
  9. 87 – 127
  10. 150 + 106

- **Oefening 3:** vul onderstaande tabel aan. Gebruik telkens een 8-bit computerwoord

getal	Teken + abs	Excess-127	2's-compl.
106			
-106			
127			
-127			
128			
-128			

1.5 Floating-point

1.5.1 Hoe kommagetallen voorstellen

In het decimaal stelsel kunnen we een kommagetal voorstellen op verschillende manieren (zie figuur 1.67).

Aangezien het grondtal 10 is voor de decimale getallen, kunnen we door gebruik te maken van machten van dit grondtal, de komma naar links of net naar rechts verschuiven. In de figuur is duidelijk te zien dat bij opschuiving van de komma naar rechts, de macht van 10 verminderd wordt met 1.

$$0{,}005 = 0{,}05 \;.\; \mathbf{10}^{-1} = 0{,}5 \;.\; \mathbf{10}^{-2} = 5 \;.\; \mathbf{10}^{-3}$$

Figuur 1.67: Verschillende voorstellingen van een kommagetal in het tiendelig talstelsel

In het binaire talstelsel kunnen we kommagetallen op dezelfde manier voorstellen als in het decimale talstelsel. Aangezien het grondtal hier 2 is, kunnen we op analoge manier de komma verschuiven op basis van de machten van 2 (zie figuur 1.68).

$$0{,}011 = 0{,}11 \,.\, 2^{-1} = 1{,}1 \,.\, 2^{-2} = 11 \,.\, 2^{-3}$$

Figuur 1.68: Verschillende voorstellingen van een kommagetal in het binair talstelsel

1.5.2 Floating-point voorstelling

De floating-point voorstelling wordt gebruikt voor de binaire voorstelling van een kommagetal in de computer.

Kijken we weer eerst hoe dit in zijn werk gaat bij decimale getallen.

Bij de voorstelling van een decimaal kommagetal is het aantal cijfers na de komma, variabel. Vandaar dat we moeten bijhouden op welke positie de komma staat in het decimale kommagetal. We gaan bijgevolg twee getallen nodig hebben om één kommagetal voor te stellen, namelijk:

1. Eén getal die de opeenvolgende cijfers van het decimale getal voorstellen
2. Eén met de aanduiding waar de komma moet komen. Dit duiden we aan onder de vorm van een **exponent** van het grondtal.

Verschuiven we nu de komma tot net na het eerste of het meest linkse beduidende cijfer, dan bekomen we de **wetenschappelijke notatie**. Een voorbeeld wordt getoond in figuur 1.69.

$$123456 = 1{,}23456 \,.\, 10^{5} \text{ of } 1{,}23456^{E}05$$

Figuur 1.69: Wetenschappelijke notatie in het decimaal talstelsel

Gaan we nu over naar de binaire voorstelling van een kommagetal. Ook hier is het aantal cijfers na de komma variabel. We hebben hier echter een voordeel: in de binaire voorstelling van een kommagetal is het eerste beduidend cijfer of de msb steeds gelijk aan 1. Er zijn immers maar twee cijfers bij de binaire voorstellen, namelijk 0 en 1.

Hiervan kunnen we handig gebruik maken om de genormaliseerde vorm te bepalen: bij de genormaliseerde vorm verschuiven we de komma zodat het eerste beduidende cijfer (altijd cijfer 1) voor de komma staat, naar analogie met de wetenschappelijke notatie van de decimale getallen. Aangezien we weten dat het getal voor de komma altijd 1 is, hoeven we dit nergens meer bij te houden. In de genormaliseerde vorm hebben we dan nog:

1. Eén getal nodig om de **mantisse** (of het cijfergedeelte na de komma) te bepalen
2. Eén getal om de **exponent**, van het grondtal 2, te bepalen.

Wat bedoeld wordt met mantisse en met exponent in de genormaliseerde vorm, wordt geïllustreerd in figuur 1.70.

$$(101{,}1101)_2 = (1{,}\underbrace{011101}_{\text{mantisse}})_2 \;.\; 2^{2} \longrightarrow \text{exponent}$$

Figuur 1.70: Genormaliseerde vorm in het binair talstelsel

1.5.3 Floating-point in de computer

Bij de voorstelling van floating-point getallen in de computer zullen we moeten afspreken hoeveel bits we gebruiken voor de mantisse en hoeveel voor de exponent.

Deze aantallen zijn vastgelegd in **normen**.

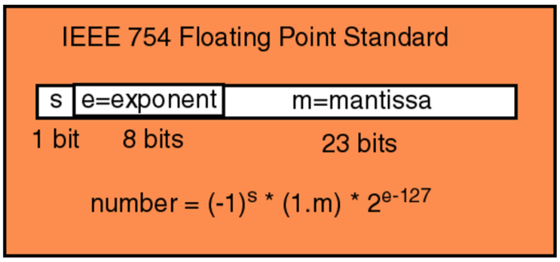
Er zijn 4 mogelijk gestandaardiseerde formaten, bepaald volgens de IEEE 754, om floating point getallen voor te stellen. Deze zijn weergegeven in figuur 1.71.

naam	precisie	sign-bit	exponent bits	mantisse bits	Excess -N	totaal
binary16	half-precision	1	5	10	15	16
binary32	single-precision	1	8	23	127	32
binary64	double-precision	1	11	52	1023	64
binary128	quadruple-precision	1	15	112	16383	128

Figuur 1.71: IEEE 754 gestandaardiseerde formaten voor floating point voorstellingen

We zullen ons bij de verdere bespreking beperken tot het binary32-formaat of enkelvoudige precisie.

Een IEEE 754 getal met enkelvoudige precisie bestaat uit 32 bit en heeft de volgende opbouw (zie figuur 1.72):



Figuur 1.72: Opbouw van een IEEE 754 getal met enkelvoudige precisie

1. De eerste bit (s) geeft weer of het getal positief (bitwaarde = 0) of negatief (bitwaarde = 1) is.
2. De volgende 8 bits zijn voor de exponent (e) die positief of negatief kan zijn en voorgesteld wordt met de excess-N methode. N = 127 (zie figuur 1.71) voor een binary32 getal.
3. De laatste 23 bits zijn voor de mantisse (m).

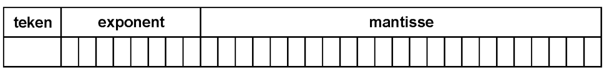
In de volgende twee paragrafen bespreken we de omzetting van een IEEE 754 binary32 getal naar zijn decimale waarde en omgekeerd.

1.5.4 Decimale waarde van een IEEE 754 binary32 getal

Om de omzetting te illustreren maken we gebruik van het volgende voorbeeld: gegeven is het volgend binary32 getal:

11000000101100000000000000000000

Om de verschillende onderdelen van een binary32 getal gemakkelijker te onderscheiden maken we gebruik van het sjabloon in figuur 1.73.



Figuur 1.73: Sjabloon voor omzetting van IEEE 754 binary32 getal

We vullen in dit sjabloon het om te zetten binary32 getal in (zie figuur 1.74).

Welke decimale waarde stelt dit nu voor?

teken	exponent	mantisse
1	1 0 0 0 0 0 0 1	0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Figuur 1.74: Voorbeeld omzetting van IEEE 754 binary32 getal

1. In het hokje onder teken is er plaats voor 1 bit. Dit stelt dus het teken (of s uit figuur 1.72) voor. De waarde van deze bit is in ons voorbeeld 1. Dit houdt dus in dat het getal negatief is. Was deze bit 0 dan ging het over een positief getal.

2. De volgende 8 bits vormen de exponent (e) voorgesteld in de excess-127. We zetten eerst de binaire waarde 10000001 om in zijn overeenkomstige decimale waarde. We krijgen de waarde 129. Deze waarde verminderen we nu nog met 127 of dus 129-127, hetgeen een waarde 2 oplevert voor de exponent.

3. De laatste 23 bits geven de binaire waarde voor de mantisse (m) of dus m = 011000. De mantisse is enkel het deel na de komma. Inclusief het getal voor de komma wordt dit 1,011000. Zetten we dit om naar de overeenkomstige decimale waarde, dan krijgen we het getal 1,375.

De uiteindelijke decimale waarde van het gegeven IEEE 754 binary32 getal wordt weergegeven in figuur 1.75.

$$-1{,}375 \;.\; 2^2 = (-5{,}5)_{10}$$

Figuur 1.75: Omzetting van IEEE 754 binary32 naar decimaal

1.5.5 IEEE 754 binary32 getalwaarde van een decimaal getal

Opnieuw wordt de omzetting geïllustreerd aan de hand van een voorbeeld.

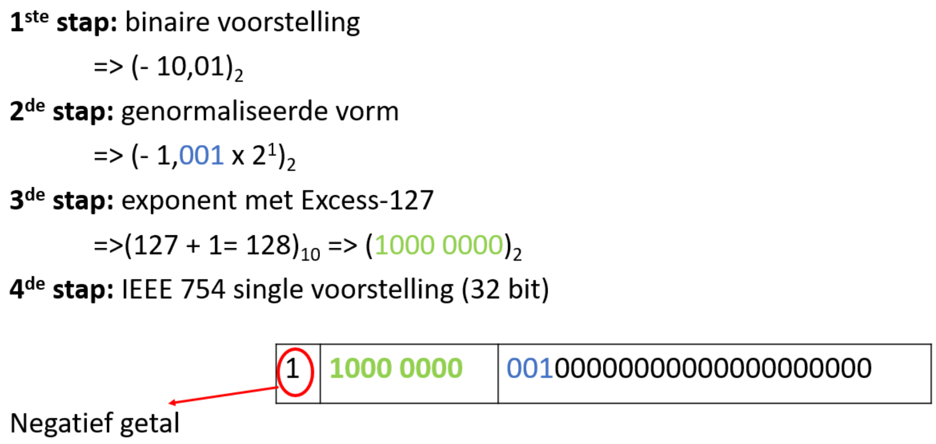
de IEEE 754 binary32 getalwaarde wordt gezocht van het volgende decimaal getal:

$$(-2{,}25)_{10}$$

In figuur 1.76 worden de verschillende stappen van de omzetting getoond:

Zoals te zien is in figuur 1.76 bestaat de omzetting uit 3 stappen:

1. In de eerste stap wordt het decimaal getal omgezet naar zijn binaire voorstelling, zoals eerder aangeleerd in dit hoofdstuk. We behouden het minteken, om aan te duiden dat het over een negatief getal gaat.



Figuur 1.76: Stappen voor de omzetting van 2,25 naar de overeenkomstige IEEE 754 binary32-waarde

2. Het binair getal uit stap 1 wordt omgezet in zijn genormaliseerde vorm. Dit betekent dat de komma verschoven wordt, zodat maar één getal voor de komma staat. Omdat het gaat over een binair getal, is dit het getal 1. Bij verschuiving van de komma, moeten we, om dezelfde getalwaarde te behouden, het bekomen getal vermenigvuldigen met een macht van 2.

   - Bij het naar links verschuiven van de komma wordt per opgeschoven positie de exponent van 2 vermeerderd met 1.
   - Bij het naar rechts verschuiven van de komma wordt per opgeschoven positie de exponent van 2 verminderd met 1.

   In het geval van ons voorbeeld in figuur 1.76, wordt de komma één positie naar links opgeschoven. Dit betekent dat de exponent van 2 de waarde 1 heeft.

3. Vervolgens wordt de Excess-127-waarde van de exponent bepaald. Hierbij wordt 127 vermeerderd met de waarde van de exponent bepaald in stap 2. De bekomen waarde wordt omgezet naar zijn binaire waarde in een 8 bits voorstelling.

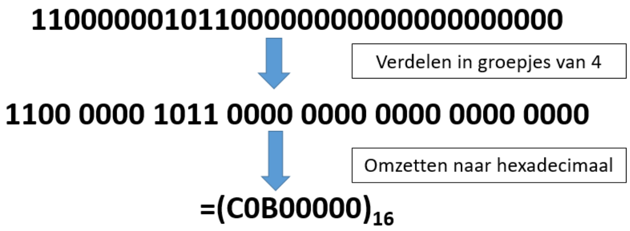
4. Tenslotte wordt alles ingevuld in het sjabloon (zie figuur 1.73):

   - In het eerste vakje wordt een 0 of een 1 geplaatst, al naargelang het getal positief of negatief is.
   - De volgende 8 bits stellen de Excess-127 binaire voorstelling voor van de exponent. Deze waarde is terug te vinden in stap 3. Zie het groene gedeelte in figuur 1.76.
   - Als laatste wordt de mantisse ingevuld. De eerste bits van de mantisse, worden teruggevonden in stap 2 (zie het blauwe deel in figuur 1.76). Verder wordt het mantisse-gedeelte aangevuld met nullen.

1.5.6 Hexadecimale voorstelling van een IEEE 754 binary32 getal

Doordat een binary32 getal nogal lang is, kan er bij het overnemen van het getal redelijk gemakkelijk fouten gemaakt worden.

Dit probleem wordt opgelost door gebruik te maken van de hexadecimale voorstelling. Hiertoe wordt het binary32 getal in groepjes van 4 bits verdeeld en vervolgens wordt elk groepje van 4 bits omgezet in zijn hexadecimale waarde. In figuur 1.77 wordt dit toegepast op het voorbeeld uit paragraaf 1.5.4.



Figuur 1.77: Omzetting van IEEE 754 binary32-waarde van binair naar hexadecimaal

1.5.7 Oefeningen op floating-point

- **Oefening 1:** bepaal de decimale waarde van het volgende binary32 getal in hexadecimale vorm $(41E00000)_{16}$. Maak hierbij gebruik van onderstaand sjabloon.

teken	exponent	Mantisse

Teken = ……… => ………………………………

Exponent = $(\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots)_2$

= $(\ldots\ldots\ldots\ldots\ldots\ldots)_{10}$

=> …………………… = ……………………

Mantisse = $(\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots)_2$

=> ……………………………………………

Het decimale getal is ……………………………………………

- **Oefening 2:** Bepaal voor volgende hexadecimale getallen de decimale waarde van hun floating-point voorstelling met enkelvoudige precisie (gebruik hiervoor het sjabloon uit oefening 1):

  1) $(CE100000)_{16}$ = ……………………………………

  2) $(43C00000)_{16}$ = ……………………………………

  3) $(3FC00000)_{16}$ = ……………………………………

  4) $(42F48000)_{16}$ = ……………………………………

- **Oefening 3:** Bepaal voor volgende decimale waarden hun floating-point voorstelling met enkelvoudige precisie, in hexadecimale vorm:

  $(149{,}25)_{10}$ =

  $(-1)_{10}$ =

  $(538{,}625)_{10}$ =

  $(-5/32)_{10}$ =

HOOFDSTUK **2**

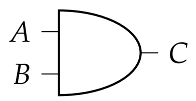
Logische Poorten

Logische poorten vormen de basis van van alle huidige computerhardware. Ze zijn een toepassing van de concepten van de Boole-algebra, die we wiskundig benaderen in hoofdstuk 3. In de praktijk wordt Boole-algebra gebruikt als basis voor het samenstellen van elektrische logische schakelingen en voor het opbouwen van computercomponenten zoals CPU's en geheugenchips. Digitale systemen gebruiken immers elektrische signalen om data op te slaan en te verwerken. Een elektrisch signaal heeft twee mogelijke toestanden: uit of aan, wat sterk overeenkomt met de constanten 0 en 1 in Boolese-algebra. Bijvoorbeeld, een elektrisch signaal tussen 0 en 0.5 Volt wordt als een 0 beschouwd en een signaal tussen 5 en 5.5 Volt als een 1. De exacte voltages kunnen verschillen van systeem tot systeem.

2.1 Overzicht van een logische poort

Een *logische poort* krijgt een aantal elektrische signalen binnen en vormt deze om tot een enkel elektrisch signaal. De binnenkomende signalen worden *ingangen* of *inputs* genoemd, het uitgaande signaal de *uitgang* of *output*.

Het gedrag van logische poorten kan beschreven worden aan de hand van waarheidstabellen. Hiervoor wordt een tabel opgesteld met een kolom voor elke mogelijke ingang en een kolom voor de uitgang. Voor elke mogelijke combinatie aan ingangswaarden wordt er een rij aan de tabel toegevoegd. Voor een poort met x ingangen en 1 uitgang heeft de waarheidstabel $x + 1$ kolommen en 2^x rijen (de titelrij niet inbegrepen). De waarheidstabel voor de logische poort in Figuur 2.1 is weergegeven in Tabel 2.1. De



Figuur 2.1: Een voorbeeld van een logische poort.

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

Tabel 2.1: De waarheidstabel voor de poort in Figuur 2.1

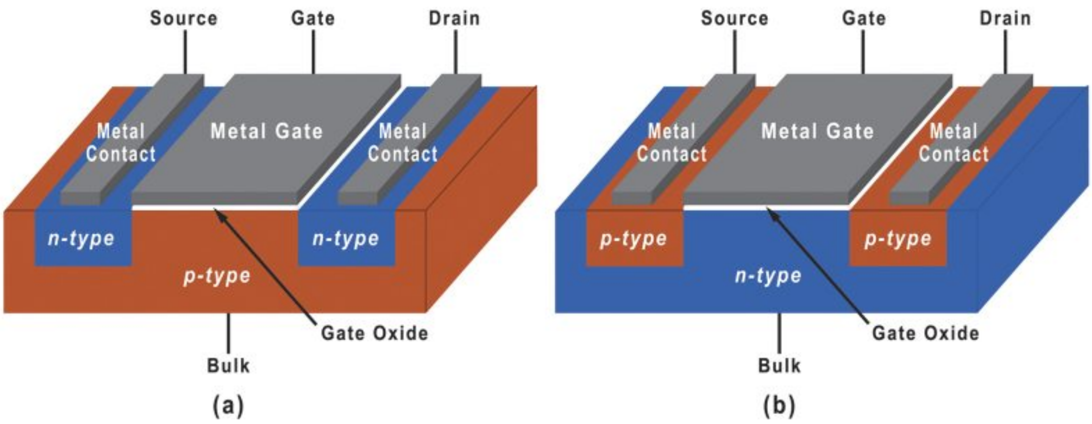
poort heeft 2 ingangen en 1 uitgang. De waarheidstabel heeft dus in dit geval 3 kolommen en 4 rijen (de titelrij niet inbegrepen).

Een poort wordt opgebouwd door elektrische componenten. De elektrische componenten die gebruikt worden om logische poorten te maken zijn *transistoren*. Een logische poort kan tot een tiental transistoren bevatten. Er zijn verschillende soorten transistoren (BJT, MOSFET, . . .). Tegenwoordig worden de meeste processoren gemaakt met CMOS technologie uit MOSFET transistoren. Hiervan bestaan er 2 types: NMOS en PMOS transistoren. Figuur 2.2 toont de elektrische symbolen voor NMOS en PMOS transistoren. Transistoren zijn zodanig klein dat ze niet te zien zijn met het blote oog. De fysische opbouw van NMOS en PMOS transistoren wordt getoond in Figuur 2.3.

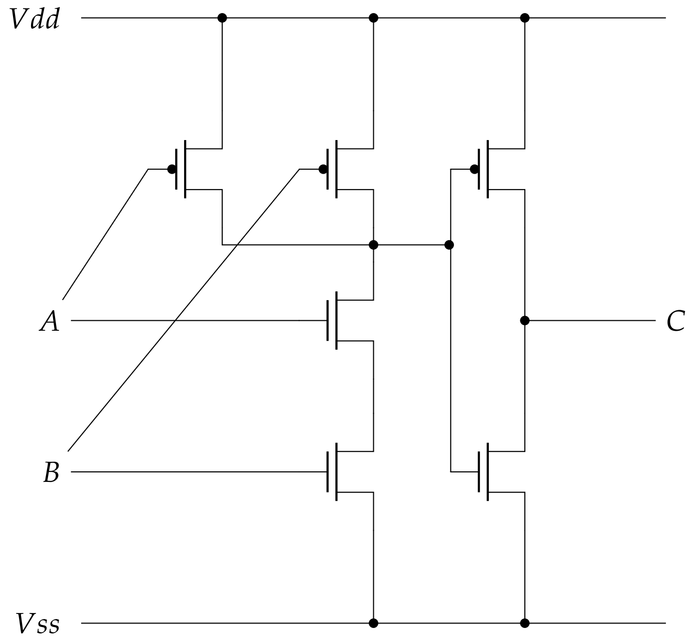
Met transistoren worden dus logische poorten gemaakt. Bijvoorbeeld, het elektrisch circuit opgebouwd uit transistoren voor de logische poort in Figuur 2.1 wordt weergegeven in Figuur 2.4. Met de logische poorten kunnen dan de logische schakelingen worden gemaakt voor de CPU, RAM en andere onderdelen van een computer.

(b) (a)

Figuur 2.2: Het symbool voor een NMOS (a) en een PMOS (b) transistor.



Figuur 2.3: De opbouw van een NMOS (a) en een PMOS (b) transistor.



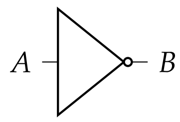
Figuur 2.4: Het elektronisch circuit voor de logische poort in Figuur 2.1.

2.2 De basispoorten

Deze sectie geeft een overzicht van de basispoorten. Deze poorten zijn de bouwstenen voor alle mogelijke digitale schakelingen.

2.2.1 De NOT-poort

De NOT-poort is de eenvoudigste met slechts één ingang en één uitgang. De NOT-poort inverteert het binnenkomende signaal. Met andere woorden: een 0 wordt een 1 en een 1 wordt een 0.



Figuur 2.5: Een NOT-poort.

A	B
0	1
1	0

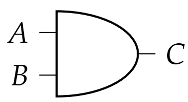
Tabel 2.2: De waarheidstabel voor een NOT-poort.

De wiskundige notatie voor een NOT-poort wordt weergegeven in (2.1).

$$\overline{A} = B \tag{2.1}$$

2.2.2 De AND-poort

De AND-poort is een poort met twee ingangen die enkel een 1 als uitgang heeft wanneer beide ingangen op 1 staan. In alle andere gevallen heeft de poort een 0 als uitgang.



Figuur 2.6: Een voorbeeld van een AND-poort.

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

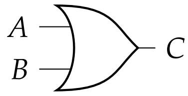
Tabel 2.3: De waarheidstabel voor een AND-poort.

De wiskundige notatie voor een AND-poort wordt weergegeven in (2.2).

$$A \cdot B = AB = C \tag{2.2}$$

2.2.3 De OR-poort

De OR-poort is een poort met twee ingangen die een 1 als uitgang heeft zodra er minstens een van de ingangen op 1 staat. Enkel als beide ingangen op 0 staan heeft de poort een 0 als uitgang.



Figuur 2.7: Een voorbeeld van een OR-poort.

A	B	C
0	0	0
0	1	1
1	0	1
1	1	1

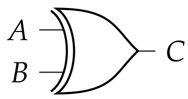
Tabel 2.4: De waarheidstabel voor een OR-poort.

De wiskundige notatie voor een OR-poort wordt weergegeven in (2.3).

$$A + B = C \tag{2.3}$$

2.2.4 De XOR-poort

De XOR-poort is een poort met twee ingangen die enkel een 1 als uitgang heeft wanneer beide ingangen van elkaar verschillen. Als beide ingangen dezelfde waarde hebben heeft de poort een 0 als uitgang.



Figuur 2.8: Een voorbeeld van een XOR-poort.

A	B	C
0	0	0
0	1	1
1	0	1
1	1	0

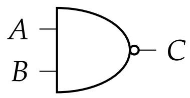
Tabel 2.5: De waarheidstabel voor een XOR-poort.

De wiskundige notatie voor een XOR-poort wordt weergegeven in (2.4).

$$\overline{A} \cdot B + A \cdot \overline{B} = C \tag{2.4}$$

2.2.5 De NAND-poort

De NAND-poort is een poort met twee ingangen die enkel een 0 teruggeeft als beide ingangen gelijk zijn aan 1. In alle andere gevallen wordt er een 1 teruggegeven.



Figuur 2.9: Een voorbeeld van een NAND-poort.

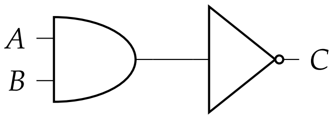
A	B	C
0	0	1
0	1	1
1	0	1
1	1	0

Tabel 2.6: De waarheidstabel voor een NAND-poort.

De wiskundige notatie voor een NAND-poort wordt weergegeven in (2.5).

$$\overline{A \cdot B} = \overline{AB} = C \tag{2.5}$$

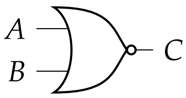
Dit komt overeen met een AND-poort gevolgd door een NOT-poort.



Figuur 2.10: Een NAND-poort gedraagt zich exact hetzelfde als een AND-poort gevolgd door een NOT-poort.

2.2.6 De NOR-poort

De NOR-poort is een poort met twee ingangen die enkel een 1 teruggeeft als beide ingangen op 0 staan. In alle andere gevallen wordt er een 0 teruggegeven.



Figuur 2.11: Een voorbeeld van een NOR-poort.

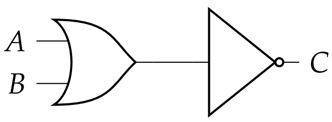
A	B	C
0	0	1
0	1	0
1	0	0
1	1	0

Tabel 2.7: De waarheidstabel voor een NOR-poort.

De wiskundige notatie voor een NOR-poort wordt weergegeven in (2.6).

$$\overline{A + B} = C \tag{2.6}$$

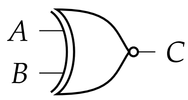
Dit komt overeen met een OR-poort gevolgd door een NOT-poort.



Figuur 2.12: Een NOR-poort gedraagt zich exact hetzelfde als een OR-poort gevolgd door een NOT-poort.

2.2.7 De XNOR-poort

De XNOR-poort is een poort met twee ingangen die een 1 teruggeeft als beide ingangen dezelfde waarde hebben. In de andere gevallen wordt er een 0 teruggegeven.



Figuur 2.13: Een voorbeeld van een XNOR-poort.

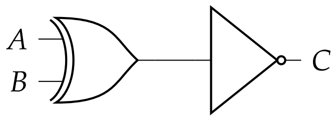
A	B	C
0	0	1
0	1	0
1	0	0
1	1	1

Tabel 2.8: De waarheidstabel voor een XNOR-poort.

De wiskundige notatie voor een XNOR-poort wordt weergegeven in (2.7).

$$\overline{\overline{A} \cdot B + A \cdot \overline{B}} = C \tag{2.7}$$

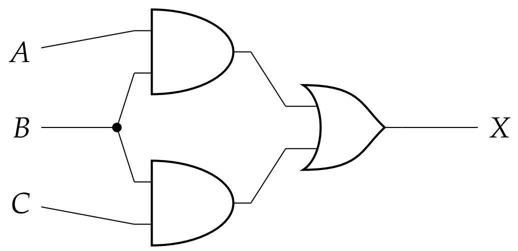
Dit komt overeen met een XOR-poort gevolgd door een NOT-poort.



Figuur 2.14: Een XNOR-poort gedraagt zich exact hetzelfde als een XOR-poort gevolgd door een NOT-poort.

2.3 Logische schakelingen

Zoals al te zien bij Figuur 2.10 en 2.12, kunnen meerdere logische poorten met elkaar verbonden worden tot een logische schakeling. De elektrische signalen worden telkens doorgegeven van poort naar poort. Figuur 2.15 toont een dergelijke logische schakeling. Kruispunten van lijnen met een zwarte bol duiden aan dat de signalen daar verbonden zijn. Als er geen zwarte bol (•) staat op het kruispunt zijn de signalen niet verbonden: ze passeren elkaar dan simpelweg. Anders gezegd, deze kruisende lijnen zijn een artifact van de tekening en hebben verder geen betekenis.



Figuur 2.15: Een logische schakeling opgebouwd uit meerdere logische poorten.

Om de waarheidstabel van een dergelijke schakeling op te stellen, volgen we signalen van de ingangen tot de uitgang. In dit voorbeeld zijn er 3 ingangen: A, B en C. Elke ingang kan ofwel de waarde 0, ofwel de waarde 1 aannemen. Dit zorgt voor 2^3 mogelijke ingangswaarden. De finale waarheidstabel heeft ook 4 kolommen: A, B, C en X. Om de signalen beter te kunnen volgen van ingang tot uitgang doorheen de schakeling nemen we ook de tussensignalen AB en BC op in de tabel. De tabel met alle mogelijke ingangswaarden is weergegeven in Tabel 2.9. Het is sterk aangeraden om deze altijd in te vullen in oplopende binaire volgorde.

A	B	C	AB	BC	X
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

Tabel 2.9: De waarheidstabel voor Figuur 2.15 met alle mogelijke ingangswaarden.

Met behulp van de waarheidstabel voor een AND-poort (Tabel 2.3) kunnen we de waarden van AB en BC (Tabel 2.10) berekenen.

A	B	C	AB	BC	X
0	0	0	0	0	
0	0	1	0	0	
0	1	0	0	0	
0	1	1	0	1	
1	0	0	0	0	
1	0	1	0	0	
1	1	0	1	0	
1	1	1	1	1	

Tabel 2.10: De waarheidstabel voor Figuur 2.15 met de waarden voor AB en BC.

Nu we de waarden weten voor AB en BC kunnen we deze samen gebruiken met de waarheidstabel voor een OR-poort (Tabel 2.4) om de uitgangswaarden van de schakeling te berekenen (Tabel 2.11).

A	B	C	AB	BC	X
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	0	0
0	1	1	0	1	1
1	0	0	0	0	0
1	0	1	0	0	0
1	1	0	1	0	1
1	1	1	1	1	1

Tabel 2.11: De waarheidstabel voor de logische schakeling in Figuur 2.15 met de uitgangswaarden.

De signalen "stromen" dus eigenlijk van de ingang van de schakeling door de poorten naar de uitgang. In Figuur 2.15 kwam dit overeen met van links naar rechts.

2.3.1 Het verbinden van poorten

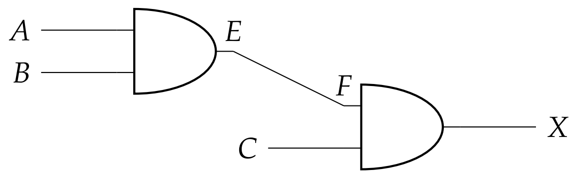
Poorten kunnen op verschillende manieren met elkaar verbonden worden in een logische schakeling. De uitgang van de ene poort kan verbonden worden aan de ingang van een andere poort om zo signalen door te geven. Signalen binnen een circuit kunnen vertakt worden om meerdere poorten te verbinden. Wel zijn niet alle verbindingen zomaar mogelijk; sommige verbindingen kunnen binnen een elektrische schakeling zelfs een kortsluiting veroorzaken. Tabel 2.12 biedt een overzicht van de mogelijke verbindingen en de eventueel daarbij optredende problemen. Deze worden in dit hoofdstuk verder besproken.

Van	Naar	Gevolg
1 poort	1 poort	Geen probleem
1 poort	Meerdere poorten	Geen probleem
Meerdere poorten	1 poort	Kortsluiting
Niets	1 of meerdere poorten	Onzeker signaal

Tabel 2.12: Een overzicht van hoe poorten verbonden kunnen worden en de mogelijke problemen die daarbij kunnen optreden.

Van 1 poort naar 1 poort

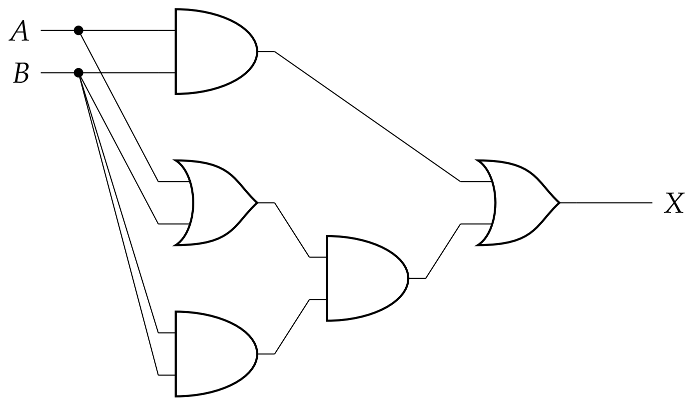
Het verbinden van de uitgang van de ene poort naar de ingang van een andere poort is geen enkel probleem. Zo gaat het signaal in Figuur 2.16 zonder problemen van de uitgang van de ene poort (E) naar de ingang van de andere poort (F).



Figuur 2.16: Een logische schakeling met een 1 poort naar 1 poort verbinding.

Van 1 poort naar meerdere poorten

De uitgang van een enkele poort verbinden met meerdere poorten is ook geen enkel probleem.



Figuur 2.17: Een logische schakeling met een 1 poort naar meerdere poorten verbinding. In dat geval worden de ingangen A en B elk verbonden met 3 poorten.

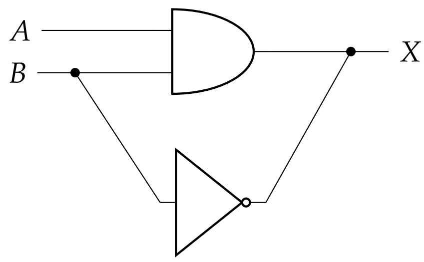
Van niets naar 1 of meerdere poorten

Wanneer een ingang van een poort niet verbonden is, is het onzeker welke waarde er op die ingang staat. Het is dan ook onzeker hoe de poort daarop zal reageren: de uitgangswaarde van de poort is dan ook onzeker. Wanneer dit voorkomt in een schakeling is het niet mogelijk om de werkelijke uitgangswaarde te achterhalen. Een onzekere waarde wordt in de waarheidstabel aangeduid met een minteken (-).

Van meerdere poorten naar 1 poort

Het verbinden van meerdere poorten naar een enkele ingang kan voor een kortsluiting zorgen. Figuur2.18 toont een schakeling met een dergelijke verbinding. In deze schakeling zijn zowel de AND- als de NOT-poort met X en dus met elkaar verbonden. Welke waarde heeft X als de uitgangswaarde van de AND-poort en de NOT-poort verschillen? Stel dat de AND-poort uitgangswaarde 0 heeft en de NOT-poort uitgangswaarde 1, wat is dan de waarde van X?

Daarnaast is dit ook gevaarlijk voor het elektrisch circuit binnenin de logische poorten. Elektrische stroom vloeit steeds van een hoger voltage naar een lager voltage. Als de uitgangswaarden van de AND- en NOT-poort verschillen, betekent dit dat er een verschillende voltage op de uitgangen van deze poorten. Hierdoor kan de elektrische stroom vloeien op een manier die niet bedoeld is en kortsluiting veroorzaken. Laten we als voorbeeld nemen dat een 1 overeenkomt met een voltage van 5 V en een 0 met een voltage van 0 V. Als de AND-poort uitgangswaarde 0 heeft en de NOT-poort uitgangswaarde 1, dan is er aan de uitgang van de NOT-poort een hoger voltage dan aan de uitgang van de AND-poort. De elektrische stroom zal dan van de uitgang van de NOT-poort vloeien naar de uitgang van de AND-poort en zo omgekeerd dan bedoeld de AND-poort binnengaan. Dit kan de AND-poort kortsluiten en eventueel schade toebrengen. Een kortsluiting wordt in de waarheidstabel aangeduid door een kruis (x).



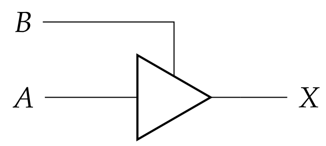
Figuur 2.18: Een logische schakeling met een meerdere poorten naar 1 poort verbinding.

A	B	AB	$\overline{B}$	X
0	0	0	1	x
0	1	0	0	0
1	0	0	1	x
1	1	1	0	x

Tabel 2.13: De waarheidstabel voor Figuur 2.18.

2.3.2 De tri-state buffer

Om kortsluiting te vermijden, wordt er gebruik gemaakt van een *tri-state buffer*. Dit is een speciale poort met twee ingangen: een signaal en een *stuurlijn*. Figuur 2.19 toont het symbool van een tri-state buffer. Ingang A komt overeen met het signaal en ingang B met de stuurlijn. De stuurlijn bepaalt hoe het signaal wordt doorgegeven aan de uitgangswaarde van de tri-state buffer. Als de stuurlijn de waarde 1 heeft, komt wordt het signaal (A) zonder wijziging doorgegeven aan de uitgang (X). Als de stuurlijn de waarde 0 heeft, wordt de uitgang hoog-Ohmig. De weerstand van de uitgang wordt dan zo groot alsof het lijkt dat de poort daar afgeknipt is van het elektrisch circuit. Dit komt dan overeen met een onzekere waarde. Let op! De onzekere waarde komt niet overeen met een 0 of 1! Een 0 of 1 komt immers overeen met een bepaald voltage, bij een onzekere waarde weten we deze immers niet. Het komt overeen met een "gat" in het elektrisch circuit.

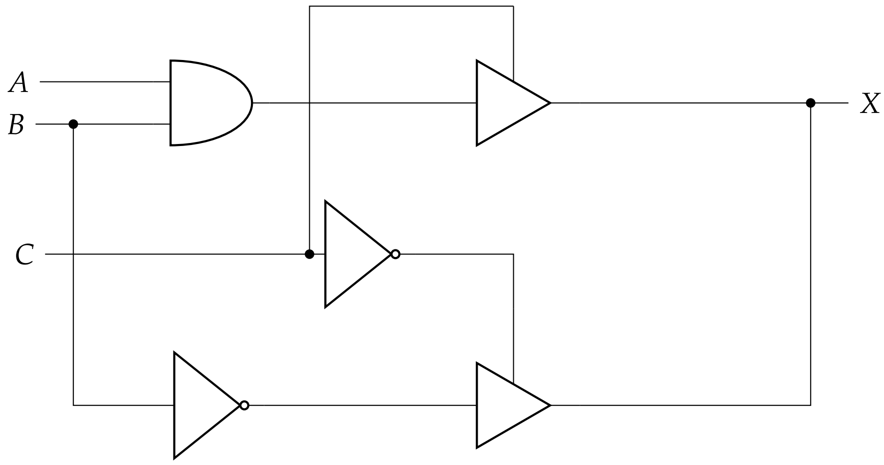


Figuur 2.19: Een logische schakeling met een meerdere poorten naar 1 poort verbinding.

A (signaal)	B (stuurlijn)	X
0	0	-
0	1	0
1	0	-
1	1	1

Tabel 2.14: De waarheidstabel voor een tri-state buffer.

Door het toevoegen van tri-state buffers kan de logische schakeling uit Figuur 2.18 beveiligd worden tegen kortsluiting. De beveiligde schakeling wordt weergegeven in Figuur 2.20. Er is een extra ingang (C) toegevoegd aan het systeem. $C = 1$ zorgt ervoor dat de uitgang van het circuit gelijk wordt aan AB. De stuurlijn van de bovenste tri-state buffer staat dan op 1, de stuurlijn van de onderste tri-state buffer op 0. De NOT-poort $\overline{B}$ is dus afgesneden van het netwerk. In het geval van $C = 0$ wordt de uitgang van het circuit gelijk aan $\overline{B}$. De stuurlijn van de bovenste tri-state buffer staat dan op 0, de stuurlijn van de onderste tri-state buffer op 1. Nu is de AND-poort AB dus afgesneden van het netwerk. AB en $\overline{B}$ zijn dus nooit tegelijkertijd aangekoppeld op het netwerk. Er kan dus geen kortsluiting optreden.



Figuur 2.20: Een logische schakeling met een meerdere poorten naar 1 poort verbinding. Dit circuit is beveiligd tegen kortsluiting dankzij de tri-state buffers.

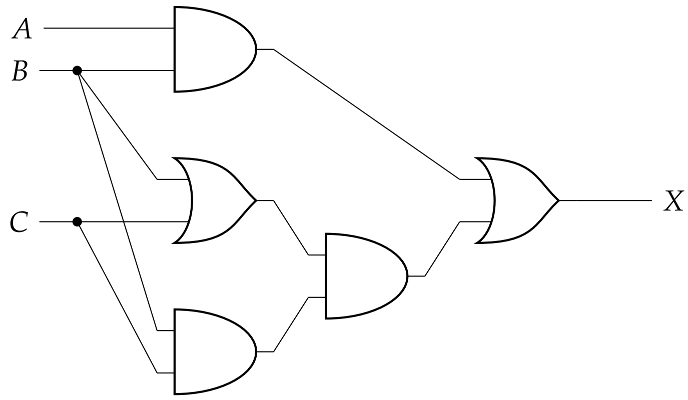
A	B	C	AB	$\overline{B}$	$\overline{C}$	Bovenste TSB	Onderste TSB	X
0	0	0	0	1	1	-	1	1
0	0	1	0	1	0	0	-	0
0	1	0	0	0	1	-	0	0
0	1	1	0	0	0	0	-	0
1	0	0	0	1	1	-	1	1
1	0	1	0	1	0	0	-	0
1	1	0	1	0	1	-	0	0
1	1	1	1	0	0	1	-	1

Tabel 2.15: De waarheidstabel voor Figuur 2.20.

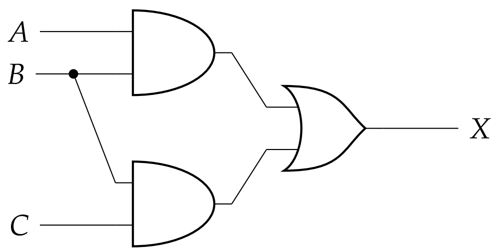
2.4 Equivalente schakelingen

Equivalente schakelingen zijn schakelingen die voor alle mogelijke inputs steeds dezelde outputs geven. Equivalente schakelingen hebben dus exact dezelfde waarheidstabel. Equivalente schakelingen kunnen dus ongemerkt door elkaar vervangen worden. Om de kosten en oppervlakte van eletronica zo klein mogelijk te houden, wordt er steeds getracht om een equivalente schakeling te vinden met zo weinig mogelijk logische poorten. Karnaugh diagrammen kunnen hierbij helpen (zie hoofdstuk 5).

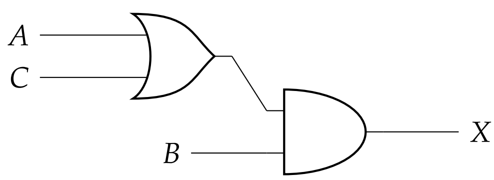
Figuur 2.21, 2.22 en 2.23 bijvoorbeeld zijn allemaal equivalent. De waarheidstabellen worden getoond in respectievelijk Tabel 2.16, 2.17 en 2.18. De waarheidstabellen geven voor alle equivalente circuits voor een bepaalde input steeds dezelfde output. Zo geeft bijvoorbeeld $A = 1$, $B = 0$, $C = 1$ voor elk circuit steeds als output 0 en $A = 1$, $B = 1$, $C = 0$ als output 1.



Figuur 2.21: $X = AB + (B + C)BC$



Figuur 2.22: $X = AB + BC$



Figuur 2.23: $X = B(A + C)$

A	B	C	AB	$B+C$	BC	$(B+C)BC$	X
0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0
0	1	0	0	1	0	0	0
0	1	1	0	1	1	1	1
1	0	0	0	0	0	0	0
1	0	1	0	1	0	0	0
1	1	0	1	1	0	0	1
1	1	1	1	1	1	1	1

Tabel 2.16: De waarheidstabel voor Figuur 2.21).

A	B	C	AB	BC	X
0	0	0	0	0	0
0	0	1	0	0	0
0	1	0	0	0	0
0	1	1	0	1	1
1	0	0	0	0	0
1	0	1	0	0	0
1	1	0	1	0	1
1	1	1	1	1	1

Tabel 2.17: De waarheidstabel voor Figuur 2.22).

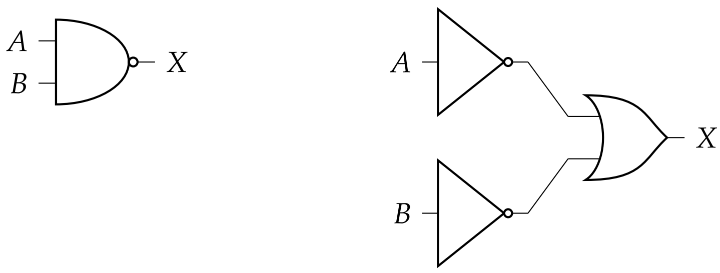
A	B	C	$A+C$	X
0	0	0	0	0
0	0	1	1	0
0	1	0	0	0
0	1	1	1	1
1	0	0	1	0
1	0	1	1	0
1	1	0	1	1
1	1	1	1	1

Tabel 2.18: De waarheidstabel voor Figuur 2.23).

Een mooi voorbeeld van equivalente schakelingen zijn de wetten van De Morgan (zie hoofdstuk 3.3.8) gegegeven in (2.8) en (2.9). Telkens is de schakeling dat het linkergedeelte van een wet van De Morgan voorstelt equivalent aan de schakeling dat het rechtergedeelte voorstelt. De waarheidstabellen van beide delen zijn gelijk.

$$\overline{AB} = \overline{A} + \overline{B} \tag{2.8}$$

$$\overline{A+B} = \overline{A} \cdot \overline{B} \tag{2.9}$$

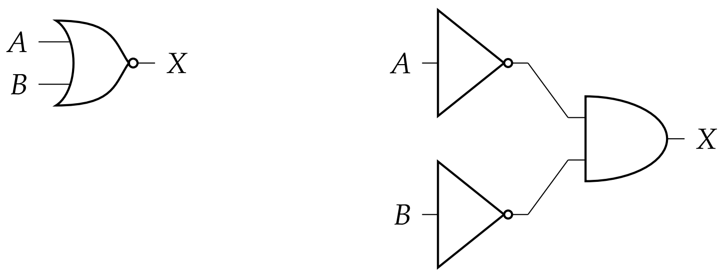


Figuur 2.24: Deze schakelingen komen overeen met respectievelijk het linker- en rechtergedeelte in 2.8).

A	B	X
0	0	1
1	0	1
0	1	1
1	1	0

A	B	$\overline{A}$	$\overline{B}$	X
0	0	1	1	1
0	1	1	0	1
1	0	0	1	1
1	1	0	0	0

Tabel 2.19: De waarheidstabel voor respectievelijk het linker- en rechtergedeelte in 2.8).



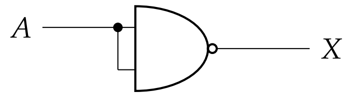
Figuur 2.25: Deze schakelingen komen overeen met respectievelijk het linker- en rechtergedeelte in 2.9).

A	B	X
0	0	1
1	0	0
0	1	0
1	1	0

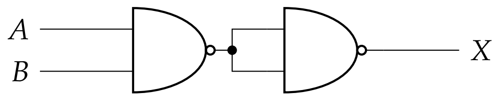
A	B	$\overline{A}$	$\overline{B}$	X
0	0	1	1	1
0	1	1	0	0
1	0	0	1	0
1	1	0	0	0

Tabel 2.20: De waarheidstabel voor respectievelijk het linker- en rechtergedeelte in 2.9).

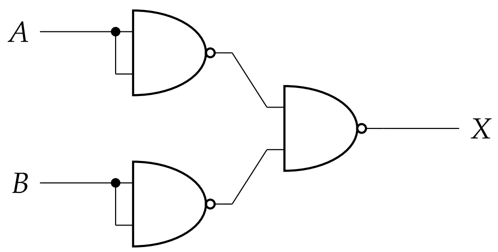
Equivalentie van schakelingen toont ook de sterkte van NAND- en NOR-poorten. Deze poorten worden namelijk ook *universele* poorten genoemd. Met behulp van enkel NAND-poorten of enkel NOR-poorten kunnen alle andere logische poorten nagemaakt worden met behulp van equivalente schakelingen. In theorie is het dus mogelijk om alle mogelijke logische schakelingen te maken met enkel NAND-poorten of enkel NOR-poorten.



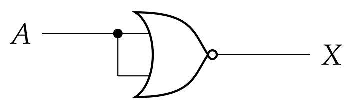
Figuur 2.26: NOT-poort opgesteld uit een NAND-poort.



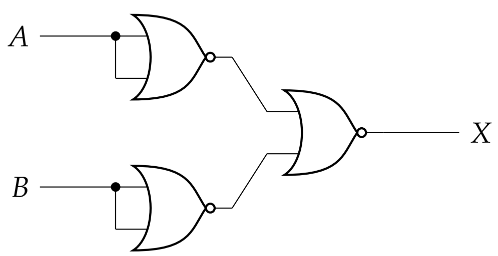
Figuur 2.27: AND-poort opgesteld uit NAND-poorten.



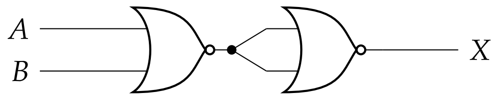
Figuur 2.28: OR-poort opgesteld uit NAND-poorten.



Figuur 2.29: NOT-poort opgesteld uit een NOR-poort.



Figuur 2.30: AND-poort opgesteld uit NOR-poorten.



Figuur 2.31: OR-poort opgesteld uit NOR-poorten.

HOOFDSTUK **3**

Boole-algebra's

In dit deel van de cursus bestuderen we de specifieke wiskundige structuur van een *Boole-algebra*.

In een eerste fase wordt de definitie van een willekeurige Boole-algebra geformuleerd samen met de belangrijkste eigenschappen van een Boole-algebra.
De definitie en eigenschappen worden geïllustreerd a.d.h.v. een aantal voorbeelden, met in het bijzonder het voorbeeld van de *minimale Boole-algebra*.

De minimale Boole-algebra vormt de basis van alle huidige computerhardware. Alle PC-schakelingen zijn opgebouwd volgens de regels van de minimale Boole-algebra en kunnen beschreven worden d.m.v. *Boolese functies en uitdrukkingen*.

Deze Boolese functies en uitdrukkingen komen in het verdere verloop aan bod.
Enerzijds wordt er gekeken hoe een Boolese functie kan herleid worden naar haar *standaardvorm*. Anderzijds wordt er aangegeven hoe een Boolese functie kan vereenvoudigd worden a.d.h.v. *Veitch Karnaugh diagrammen*.

In de cursus computerarchitectuur wordt dieper ingegaan op de praktische toepassingen van de minimale Boole-algebra en haar functies.

Over Boole-algebra's zijn zeer veel boeken geschreven. Twee werken die terug te vinden zijn in de schoolbibliotheek zijn: *Boole-algebra's* uit de reeks Opbouw [Opbouw, 1990] en *Boolese algebra's* van Bouqué [Bouqué, 1972].

3.1 Definitie

Een Boole-algebra is één specifieke algebraïsche structuur naast veel andere structuren die binnen de wiskunde gebruikt worden. Elke wiskundige structuur heeft zijn eigen regels en wetmatigheden.
We starten met het beschrijven van de structuur van een Boole-algebra.

Definitie 3.1 *Een* ***Boole-algebra*** $\mathcal{B}$ *bestaat uit*

- *een verzameling* S *die minstens twee constanten,* 0 *en* 1*, bevat;*
- *twee binaire operatoren op* S*:* $+$ *en* $\cdot$*;*
- *een unaire operator op* S*:* $^{-}$ *(complement).*

In een Boole-algebra moeten voor alle $x, y, z \in S$ *de* ***axioma's van Huntington*** *geldig zijn:*

1. *commutatieve wetten:*
$$\begin{aligned} x + y &= y + x \\ x \cdot y &= y \cdot x \end{aligned}$$
2. *distributieve wetten:*
$$\begin{aligned} x \cdot (y + z) &= (x \cdot y) + (x \cdot z) \\ x + (y \cdot z) &= (x + y) \cdot (x + z) \end{aligned}$$
3. *identiteitswetten:*
$$\begin{aligned} x + 0 &= x \\ x \cdot 1 &= x \end{aligned}$$
4. *complementwetten:*
$$\begin{aligned} x + \overline{x} &= 1 \\ x \cdot \overline{x} &= 0. \end{aligned}$$

Notatie: $\mathcal{B} = (S, +, \cdot, ^{-}, 0, 1) = (S, +, \cdot, ^{-})$.

Let op! De wiskundige symbolen $+$, $\cdot$, 0 en 1 die hier gebruikt worden voor het definiëren van een Boole-algebra hebben NIET dezelfde betekenis als de $+$, $\cdot$, 0 en 1 die gebruikt worden binnen bv. de theorie van de reële getallen.
De betekenis die aan deze symbolen wordt meegegeven, kan voor elke Boole-algebra anders zijn. Dit wordt geïllustreerd in de voorbeelden.

Alvorens over te gaan tot de voorbeelden, maken we eerst nog een aantal praktische afspraken.
Indien de binaire operatoren van een Boole-algebra worden voorgesteld door $+$ en $\cdot$ dan houden we ons aan de volgende regels:

- de binaire operator $\cdot$ moet niet steeds expliciet opgeschreven worden, zo is $x \cdot y = xy$;
- bij de uitwerking van een uitdrukking heeft de bewerking $\cdot$ steeds voorrang op $+$ tenzij haakjes een andere volgorde aangeven.

3.1.1 Voorbeeld 1

Een Boole-algebra bestaat uit een niet lege verzameling waarop twee binaire operatoren en één unaire operator zijn gedefinieerd.
In het eerste voorbeeld wordt de verzameling S gedefinieerd als zijnde de verzameling $\mathcal{P}(\mathcal{U})$, de machtsverzameling van het universum $\mathcal{U}$. Op deze verzameling zijn de twee binaire operatoren $\cup$ (*unie*) en $\cap$ (*doorsnede*) gedefinieerd. Als unaire operator wordt gekozen voor $^{-}$ (*absoluut complement*). Voor elke niet lege verzameling $\mathcal{U}$ geldt bovendien dat $\emptyset \in \mathcal{P}(\mathcal{U})$ en $\mathcal{U} \in \mathcal{P}(\mathcal{U})$.
De definitie van deze begrippen vind je terug in deel 1 'Verzamelingen, relaties en functies'.

Is de structuur $(\mathcal{P}(\mathcal{U}), \cup, \cap, ^{-}, \emptyset, \mathcal{U})$ een Boole-algebra?

Alle componenten voor een Boole-algebra zijn aanwezig, nl.:

- een niet lege verzameling: $\mathcal{P}(\mathcal{U}) \quad (\emptyset \in \mathcal{P}(\mathcal{U}), \mathcal{U} \in \mathcal{P}(\mathcal{U}))$;
- twee binaire operatoren: $\cup$ en $\cap$;
- één unaire operator: $^{-}$.

Voldoen deze operatoren aan de axioma's van Huntington?

Stel $A, B, C \in \mathcal{P}(\mathcal{U})$. De vier axioma's herschreven voor deze specifieke situatie zijn:

1. commutatieve wetten:
$$\begin{aligned} A \cup B &= B \cup A \\ A \cap B &= B \cap A \end{aligned}$$

2. distributieve wetten:
$$\begin{aligned} A \cap (B \cup C) &= (A \cap B) \cup (A \cap C) \\ A \cup (B \cap C) &= (A \cup B) \cap (A \cup C) \end{aligned}$$

3. identiteitswetten:
$$\begin{aligned} A \cup \emptyset &= A \\ A \cap \mathcal{U} &= A \end{aligned}$$

4. complementwetten:
$$\begin{aligned} A \cup \overline{A} &= \mathcal{U} \\ A \cap \overline{A} &= \emptyset. \end{aligned}$$

Al deze axioma's zijn geldige eigenschappen binnen de verzamelingenleer. Dit wordt uitvoerig beschreven in deel 1 'Verzamelingen, relaties en functies'.

Dus, de verzameling $\mathcal{P}(\mathcal{U})$ is een niet ledige verzameling met minstens twee elementen: nl. $\emptyset$ en $\mathcal{U}$. Op deze verzameling zijn de twee binaire operatoren $\cup$ en $\cap$ gedefinieerd en de unaire operator $^-$. Voor deze bewerkingen zijn de axioma's van Huntington geldig.
Hieruit mogen we besluiten dat de structuur $(\mathcal{P}(\mathcal{U}), \cup, \cap, ^-, \emptyset, \mathcal{U})$ een Boole-algebra is.

3.1.2 Voorbeeld 2

Stel $B = \{0, 1\}$. Op deze verzameling B worden drie bewerkingen gedefinieerd: twee binaire operatoren, nl. $+$ en $\cdot$, en één unaire operator, $^-$. De definitie van deze drie bewerkingen wordt gegeven a.d.h.v. de volgende tabellen:

$+$	0	1
0	0	1
1	1	1

$\cdot$	0	1
0	0	0
1	0	1

	$^-$
0	1
1	0

Is de structuur $(B, +, \cdot, ^-, 0, 1)$ een Boole-algebra? Indien ja, dan zal dit de kleinst mogelijke Boole-algebra zijn: de verzameling B bevat juist twee elementen, het minimum aantal voor een Boole-algebra.

De opgegeven structuur voldoet aan de definitie van een Boole-algebra als de vier axioma's van Huntington voldaan zijn. We verifiëren de geldigheid van deze axioma's.

Stel $a, b, c \in B$ dan geldt:

1. **Commutativiteit:**
$$\begin{aligned} a + b &= b + a \\ a \cdot b &= b \cdot a. \end{aligned}$$

   Bewijs
   Aangezien de verzameling B maar een beperkt aantal elementen heeft, meer bepaald 2, kunnen we snel alle mogelijke combinaties voor a en b overlopen. Indien voor alle mogelijke waarden van a en b de gelijkheid geldt, is de wet bewezen.
   De volgende tabel illustreert dit voor de uitdrukking $a + b = b + a$.

a	b	$a+b$	$b+a$
0	0	0	0
0	1	1	1
1	0	1	1
1	1	1	1

   In elke mogelijke situatie is $a + b = b + a$. Dit zien we in de voorlaatste en laatste kolom.
   Dit bewijst de commutativiteit van de binaire operator $+$.

   Op analoge manier kan de commutativiteit van de binaire operator $\cdot$ geverifieerd worden.

Opmerking:
Indien de definitie-tabel van een binaire operator symmetrisch is t.o.v. de diagonaal (d.i. als we spiegelen t.o.v. de diagonaal, vallen gelijke elementen op elkaar), is de operator commutatief. Dit geeft een alternatieve en snellere manier om de commutativiteit van een binaire operator te verifiëren.

2. **Distributiviteit:**

$$\begin{aligned} a \cdot (b + c) &= (a \cdot b) + (a \cdot c) \\ a + (b \cdot c) &= (a + b) \cdot (a + c). \end{aligned}$$

Bewijs
We verifiëren de tweede gelijkheid, de werkwijze voor de eerste gelijkheid is analoog.
De variabelen a, b en c kunnen elk slechts twee waarden aannemen, nl. 0 of 1. Als we alle combinaties voor a, b en c overlopen, resulteert dit in $2 \times 2 \times 2 = 8$ verschillende situaties. Die zijn samengevat in onderstaande tabel:

a	b	c	$b \cdot c$	$a + (b \cdot c)$	$a + b$	$a + c$	$(a + b) \cdot (a + c)$
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0
0	1	0	0	0	1	0	0
0	1	1	1	1	1	1	1
1	0	0	0	1	1	1	1
1	0	1	0	1	1	1	1
1	1	0	0	1	1	1	1
1	1	1	1	1	1	1	1

In alle mogelijke situaties is het resultaat van het linkerlid gelijk aan het resultaat van het rechterlid. Dit bewijst de geldigheid van de distributieve wet.

3. **Identiteit:**

$$\begin{aligned} a + 0 &= a \\ a \cdot 1 &= a. \end{aligned}$$

Bewijs
In de tabel, die de bewerking + definieert, zien we dat het element 0 optellen bij om het even welk element a uit B geen invloed heeft op a. Dit bewijst dat 0 *neutraal element* is voor de +.
Op analoge manier lezen we uit de tabel van de bewerking $\cdot$ af dat 1 inderdaad het neutraal element voor de bewerking $\cdot$ is.
Indien gewenst kan deze wet expliciet aangetoond worden als volgt:

a	$a + 0$	$a \cdot 1$
0	0	0
1	1	1

4. **Complement:**

$$\begin{aligned} a + \overline{a} &= 1 \\ a \cdot \overline{a} &= 0. \end{aligned}$$

Bewijs
We maken opnieuw gebruik van een tabel om alle mogelijke situaties te verifiëren:

a	$\overline{a}$	$a+\overline{a}$	$a \cdot \overline{a}$
0	1	1	0
1	0	1	0

Besluit:
Voor de opgegeven structuur zijn de vier axioma's van Huntington voldaan. We mogen besluiten dat de structuur $(B, +, \cdot, ^-, 0, 1)$ een Boole-algebra is. Deze wordt de **minimale Boole-algebra** genoemd.

3.1.3 Voorbeeld 3

Stel $Prop = \{alle\ proposities\}$, waarbij een propositie een uitspraak is die enkel waar (W) of onwaar (O) kan zijn. De bewerkingen $\vee$ (*of*), $\wedge$ (*en*) , $\neg$ (*niet*) worden als volgt gedefinieerd

$\vee$	O	W
O	O	W
W	W	W

$\wedge$	O	W
O	O	O
W	O	W

	$\neg$
O	W
W	O

Analoog aan het voorgaande voorbeeld is ook deze structuur $(Prop, \vee, \wedge, \neg, O, W)$ een Boole-algebra. Deze algebra vormt de basis voor de eerste orde propositielogica.

Het bewijs kan op dezelfde manier gegeven worden als voor de minimale Boole-algebra, met dit verschil dat de binaire operatoren nu niet als + en · genoteerd worden maar als $\vee$ (of) en $\wedge$ (en). De unaire operator wordt nu voorgesteld door $\neg$ (niet).

3.1.4 Oefeningen

1. Noem D de verzameling gehele delers van 6. Definieer de volgende operatoren op D:

$$\begin{aligned} x + y &= \mathrm{kgv}(x, y) \\ x \cdot y &= \mathrm{ggd}(x, y) \\ \overline{x} &= 6/x. \end{aligned}$$

   Is $(D, +, \cdot, ^-)$ een Boole-algebra? Motiveer uw antwoord.

2. Noem D de verzameling gehele delers van 8. Definieer de volgende operatoren op D:

$$\begin{aligned} x + y &= \text{kgv}(x, y) \\ x \cdot y &= \text{ggd}(x, y) \\ \overline{x} &= 8/x. \end{aligned}$$

Is $(D, +, \cdot, ^{-})$ een Boole-algebra? Motiveer uw antwoord.

3.2 Dualiteitsbeginsel

Keren we even terug naar de definitie van een Boole-algebra. Elke wet die geldig moet zijn binnen een Boole-algebra bestaat uit twee axioma's. Deze axioma's worden elkaars *duale* genoemd.
In het bijzonder geldt dat voor elk axioma dat geldig is in een Boole-algebra ook het duale axioma geldig is in die Boole-algebra.

axioma	duale vorm
$a + b = b + a$	$a \cdot b = b \cdot a$
$a \cdot (b + c) = (a \cdot b) + (a \cdot c)$	$a + (b \cdot c) = (a + b) \cdot (a + c)$
$a + 0 = a$	$a \cdot 1 = a$
$a + \overline{a} = 1$	$a \cdot \overline{a} = 0$

Principe *Met elk axioma en elke eigenschap van een Boole-algebra* $(S, +, \cdot, ^{-}, 0, 1)$ *stemt een duaal axioma of een duale eigenschap van die Boole-algebra overeen.*

Praktisch betekent dit dat wanneer bewezen is dat een bepaalde eigenschap geldig is binnen een Boole-algebra, we zonder bewijs mogen aannemen dat ook de duale eigenschap zal geldig zijn binnen een Boole-algebra.

Het duaal axioma of de duale eigenschap bekomen we door de symbolen uit de betrokken uitdrukking om te zetten naar hun duale vorm.
Voor de algemene Boole-algebra $\mathcal{B} = (S, +, \cdot, ^{-}, 0, 1)$ gaat dit als volgt:

symbool	duale vorm
$+$	$\cdot$
$\cdot$	$+$
0	1
1	0
$^{-}$	$^{-}$

M.a.w. de binaire operatoren worden met elkaar verwisseld: een + wordt een $\cdot$ en vice versa. Analoog voor de neutrale elementen van de binaire operatoren. De overige symbolen in een uitdrukking blijven onveranderd staan.

Zo is bijvoorbeeld de duale vorm van de uitdrukking $\overline{x + y} \cdot 1$ gelijk aan $\overline{x \cdot y} + 0$.
Verklaring: de + wordt $\cdot$, de $\cdot$ wordt + en 1 wordt 0.

3.3 Eigenschappen

Uit de definitie van een Boole-algebra kunnen veel eigenschappen die geldig zijn binnen een Boole-algebra afgeleid worden. In wat nu volgt, worden de belangrijkste eigenschappen van een Boole-algebra besproken.

We zullen deze eigenschappen bewijzen voor een willekeurige Boole-algebra $\mathcal{B} = (S, +, \cdot, ^{-}, 0, 1)$. Concreet betekent dit dat deze eigenschappen geldig zijn voor elke mogelijke Boole-algebra zowel voor de minimale Boole-algebra als voor de Boole-algebra van de verzamelingen, als voor om het even welke andere Boole-algebra.

3.3.1 Eigenschap 1: het complement is uniek

Voor elke $x \in S$ bestaat er juist één $\overline{x} \in S$ zodat $x + \overline{x} = 1$ en $x \cdot \overline{x} = 0$.

Bewijs: Stel dat er nog een element $y \in S$ bestaat waarvoor $x + y = 1$ en $x \cdot y = 0$. We tonen aan dat $y = \overline{x}$.

$$
\begin{array}{rcll}
y & = & y \cdot 1 & \text{(identiteitswet)} \\
 & = & y \cdot (x + \overline{x}) & \text{(complementwet)} \\
 & = & y \cdot x + y \cdot \overline{x} & \text{(distributieve wet)} \\
 & = & x \cdot y + \overline{x} \cdot y & \text{(commutatieve wet)} \\
 & = & 0 + \overline{x} \cdot y & \text{(veronderstelling)} \\
 & = & x \cdot \overline{x} + \overline{x} \cdot y & \text{(complementwet)} \\
 & = & \overline{x} \cdot x + \overline{x} \cdot y & \text{(commutatieve wet)} \\
 & = & \overline{x} \cdot (x + y) & \text{(distributieve wet)} \\
 & = & \overline{x} \cdot 1 & \text{(veronderstelling)} \\
 & = & \overline{x} & \text{(identiteitswet)}
\end{array}
$$

Opmerking De duale uitdrukking van deze eerste eigenschap resulteert opnieuw in dezelfde uitdrukking en geeft dus geen extra informatie over een Boole-algebra. Dit geldt eveneens voor de tweede eigenschap de *involutie*.

3.3.2 Eigenschap 2: involutie

Als we de unaire operator tweemaal toepassen op een element x dan krijgen we als uitkomst opnieuw x.

$$\overline{\overline{x}} = x$$

Bewijs:
Voor het aantonen van een dergelijke gelijkheid zullen we steeds vertrekken van het linkerlid. Gebruik makend van de reeds bewezen eigenschappen en de axioma's van een Boole-algebra, willen we dan aantonen dat het linkerlid gelijk is aan het rechterlid.

$$
\begin{aligned}
\overline{\overline{x}} &= \overline{\overline{x}} \cdot 1 && \text{(identiteitswet)} \\
&= \overline{\overline{x}} \cdot (x + \overline{x}) && \text{(complementwet)} \\
&= (\overline{\overline{x}} \cdot x) + (\overline{\overline{x}} \cdot \overline{x}) && \text{(distributieve wet)} \\
&= (\overline{\overline{x}} \cdot x) + (\overline{x} \cdot \overline{\overline{x}}) && \text{(commutatieve wet)} \\
&= (\overline{\overline{x}} \cdot x) + 0 && \text{(complementwet)} \\
&= (\overline{\overline{x}} \cdot x) + (x \cdot \overline{x}) && \text{(complementwet)} \\
&= (x \cdot \overline{\overline{x}}) + (x \cdot \overline{x}) && \text{(commutatieve wet)} \\
&= x \cdot (\overline{\overline{x}} + \overline{x}) && \text{(distributieve wet)} \\
&= x \cdot (\overline{x} + \overline{\overline{x}}) && \text{(commutatieve wet)} \\
&= x \cdot 1 && \text{(complementwet)} \\
&= x && \text{(identiteitswet)}
\end{aligned}
$$

3.3.3 Eigenschap 3: complement van 0 en 1

Het complement van 0 is 1 en omgekeerd.

$$
\begin{aligned}
\overline{0} &= 1 \\
\overline{1} &= 0
\end{aligned}
$$

Opmerking Beide uitdrukking zijn elkaars duale. Indien één van beide uitdrukkingen bewezen is dan weten we uit het dualiteitsprincipe dat de andere (duale) uitdrukking ook geldig is.

Bewijs Hoewel het voldoende is slechts één van de twee uitdrukkingen te bewijzen, zullen we toch de beide bewijsjes noteren. Indien de geldigheid van de eerste uitdrukking is aangetoond, bekomt men het bewijs van de tweede uitdrukking door in het bewijs elke lijn om te zetten naar zijn duale vorm.

$$
\begin{aligned}
\overline{0} &= \overline{0} + 0 && \text{(identiteitswet)} \\
&= 1 && \text{(complementwet)}
\end{aligned}
\qquad
\begin{aligned}
\overline{1} &= \overline{1} \cdot 1 \\
&= 0
\end{aligned}
$$

3.3.4 Eigenschap 4: idempotentie

Wanneer de operator + toegepast wordt op tweemaal hetzelfde argument dan is het resultaat dat argument.
Wanneer de operator · toegepast wordt op tweemaal hetzelfde argument dan is het resultaat dat argument.

$$
\begin{aligned}
x + x &= x \\
x \cdot x &= x
\end{aligned}
$$

3.3.5 Eigenschap 5: begrenzing

Om het even welke waarde optellen bij 1 levert steeds 1.
Om het even welke waarde vermenigvuldigen met 0 levert steeds 0.

$$\begin{aligned} x + 1 &= 1 \\ x \cdot 0 &= 0 \end{aligned}$$

3.3.6 Eigenschap 6: absorptie

Een element x optellen bij een product waarin x voorkomt als factor levert steeds x.
Een element x vermenigvuldigen met een som waarin x als term voorkomt levert steeds x.

$$\begin{aligned} x + (x \cdot y) &= x \\ x \cdot (x + y) &= x \end{aligned}$$

Het bewijs van de drie voorgaande eigenschappen wordt gevraagd als oefening.
Het bewijs van de twee volgende eigenschappen valt buiten het bereik van deze cursus.
De eigenschappen zelf zijn daarom niet minder belangrijk, integendeel.

3.3.7 Eigenschap 7: associatief

Wanneer we drie elementen optellen dan kunnen we de haakjes verplaatsen zonder het resultaat te wijzigen.
Wanneer we drie elementen vermenigvuldigen dan kunnen we de haakjes verplaatsen zonder het resultaat te wijzigen.
We kunnen de haakjes dus ook weglaten omdat de uitdrukking niet verkeerd kan begrepen worden.

$$\begin{aligned} x + (y + z) &= (x + y) + z &&= x + y + z \\ x \cdot (y \cdot z) &= (x \cdot y) \cdot z &&= x \cdot y \cdot z \end{aligned}$$

3.3.8 Eigenschap 8: wetten van de Morgan

Het complement van een som is het product van de complementen.
Het complement van een product is de som van de complementen.

$$\begin{aligned} \overline{x + y} &= \overline{x} \cdot \overline{y} \\ \overline{x \cdot y} &= \overline{x} + \overline{y} \end{aligned}$$

3.4 Oefeningen

1. Geef het bewijs van de vierde, vijfde en zesde eigenschap: idempotentie, begrenzing en absorptie.
2. Hoe luiden de eigenschappen 5 en 6 voor de Boole-algebra $(\mathcal{P}(\mathcal{U}), \cup, \cap, ^{-}, \emptyset, \mathcal{U})$.
3. Stel dat $(S, +, \cdot, ^{-}, 0, 1)$ een Boole-algebra is met $x, y, z \in S$. Bepaal het complement en de duale uitdrukking van:
   a) $x + \overline{y \cdot z}$
   b) $(x \cdot y \cdot z) + (\overline{x} \cdot \overline{y} \cdot \overline{z})$.
4. Stel dat $(S, +, \cdot, ^{-}, 0, 1)$ een Boole-algebra is met $x, y, z, u \in S$.
   Toon de geldigheid van de volgende uitdrukkingen aan door te steunen op de axioma's en eigenschappen van een Boole-algebra.
   a) $(x + \overline{y}) \cdot (\overline{x} + y) = \overline{x} \cdot \overline{y} + x \cdot y$
   b) $x \cdot y + \overline{y} \cdot z + x \cdot z = x \cdot y + \overline{y} \cdot z$
   c) $x \cdot \overline{z} + \overline{x} \cdot \overline{z} + \overline{y} \cdot z = \overline{y} + \overline{z}$
   d) $x \cdot y + \overline{x \cdot \overline{y} + \overline{x} \cdot \overline{y}} = y$
   e) $(x \cdot y + x \cdot u) \cdot (z \cdot u + \overline{x} \cdot u) \cdot (\overline{y} \cdot z + \overline{x} \cdot \overline{y}) \cdot x \cdot \overline{y} = x \cdot \overline{y} \cdot z \cdot u$

3.5 Extra oefeningen

1. Stel dat $(S, +, \cdot, ^{-}, 0, 1)$ een Boole-algebra is met $x, y, z, u \in S$. Bepaal het complement en de duale uitdrukking van:
   a) $x + y + z$
   b) $(x \cdot y) + z$
   c) $x \cdot \overline{y} \cdot \overline{z} \cdot u$
   d) $(\overline{x} \cdot y) + (\overline{y} \cdot z)$
   e) $x + (y \cdot z) + (\overline{x} \cdot \overline{y} \cdot \overline{z})$
   f) $(x + \overline{y} + z) \cdot (x + y + \overline{z})$
2. Stel dat $(S, +, \cdot, ^{-}, 0, 1)$ een Boole-algebra is. Bewijs voor alle $x, y, z \in S$:
   a) $(x + y) \cdot (\overline{x} + \overline{y}) = x \cdot \overline{y} + \overline{x} \cdot y$
   b) $x \cdot \overline{y} + \overline{x} \cdot y + \overline{x} \cdot \overline{y} = \overline{x} + \overline{y}$
   c) $(x \cdot y + \overline{x} \cdot y \cdot z) \cdot (x \cdot z + \overline{x} \cdot \overline{y}) = x \cdot y \cdot z$
   d) $\overline{\overline{x} \cdot y + x \cdot \overline{y} + \overline{x} \cdot \overline{y}} = x \cdot y$
   e) $\overline{x} \cdot \overline{y} \cdot \overline{z} + \overline{x} \cdot y \cdot \overline{z} + \overline{x} \cdot \overline{y} \cdot z + \overline{x} \cdot \overline{y} \cdot \overline{z} = \overline{x} \cdot \overline{z} + \overline{x} \cdot \overline{y}$

3.6 Oplossingen

1. complement:

   a) $\overline{x} \cdot \overline{y} \cdot \overline{z}$
   b) $(\overline{x} + \overline{y}) \cdot \overline{z}$
   c) $\overline{x} + y + z + \overline{u}$
   d) $(x + \overline{y}) \cdot (y + \overline{z})$
   e) $\overline{x} \cdot (\overline{y} + \overline{z}) \cdot (x + y + z)$
   f) $(\overline{x} \cdot y \cdot \overline{z}) + (\overline{x} \cdot \overline{y} \cdot z)$

   duale uitdrukking:

   a) $x \cdot y \cdot z$
   b) $(x + y) \cdot z$
   c) $x + \overline{y} + \overline{z} + u$
   d) $(\overline{x} + y) \cdot (\overline{y} + z)$
   e) $x \cdot (y + z) \cdot (\overline{x} + \overline{y} + \overline{z})$
   f) $(x \cdot \overline{y} \cdot z) + (x \cdot y \cdot \overline{z})$

2. Steun op de eigenschappen van een Boole-algebra om het gevraagde te bewijzen.

HOOFDSTUK **4**

Boolese uitdrukkingen en functies

In het voorgaande hoofdstuk werd het voorbeeld van de minimale Boole-algebra aangehaald. Deze minimale algebra vindt o.a. zijn toepassing in het samenstellen van digitale elektronische schakelingen.

In de verdere bespreking van Boole-algebra's zullen we ons beperken tot deze specifieke Boole-algebra: de minimale Boole-algebra $\mathcal{B}$. We gebruiken de volgende notaties:

$\mathcal{B} = (B, +, \cdot, ^{-}, 0, 1)$, met $B = \{0, 1\}$ en

$+$	0	1
0	0	1
1	1	1

$\cdot$	0	1
0	0	0
1	0	1

	$^{-}$
0	1
1	0

In het algemeen stellen $a, b, c, \ldots \in B$ constanten voor terwijl $x, y, z, \ldots \in B$ variabelen voorstellen.

Binnen deze structuur zullen we functies definiëren. We bespreken enerzijds hoe deze functies kunnen herleid worden naar een standaardvorm. Anderzijds gaan we na hoe we een Boolese uitdrukking kunnen vereenvoudigen.

Zoals reeds vermeld, correspondeert met elke Boolese functie uit de minimale Boole-algebra een elektronische schakeling. Deze toepassing wordt verder uitgediept in de cursus computerarchitectuur.

4.1 Boolese uitdrukkingen

Definitie 4.1 *Een **Boolese functie** is van de vorm*

$$f : B^n \to B : (x_1, x_2, \ldots, x_n) \mapsto f(x_1, x_2, \ldots, x_n)$$

*met $f(x_1, x_2, \ldots, x_n)$ een **Boolese uitdrukking**.*

Boolese functies stellen we voor a.d.h.v. hun corresponderende Boolese uitdrukking of a.d.h.v. de bijhorende tabel.

Voorbeelden

1. Zij $f : B^2 \to B : (x, y) \mapsto f(x, y) = x + \overline{y} + \overline{x} \cdot y$.

   De corresponderende tabel is:

x	y	$\overline{x}$	$\overline{y}$	$\overline{x} \cdot y$	$f(x,y)$
0	0	1	1	0	1
0	1	1	0	1	1
1	0	0	1	0	1
1	1	0	0	0	1

2. Beschouw de functie $f : B^3 \to B : (x, y, z) \mapsto x \cdot (\overline{y} + z)$.

   In dit geval geldt $f(x, y, z) = x \cdot (\overline{y} + z)$.

   Aangezien f een functie is in 3 variabelen x, y en z zijn er in het totaal 8 ($= 2 \times 2 \times 2$) mogelijke inputcombinaties. Deze 8 situaties worden opgesomd in de tabel. We noteren eerste de vier mogelijkheden met $x = 0$. Van deze vier zal y twee keer 0 en twee keer 1 zijn. Voor x en y 0 kan z 0 of 1 zijn, enz..

   De corresponderende tabel is:

x	y	z	$\overline{y}$	$\overline{y} + z$	$x \cdot (\overline{y} + z)$
0	0	0	1	1	0
0	0	1	1	1	0
0	1	0	0	0	0
0	1	1	0	1	0
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	0	0	0
1	1	1	0	1	1

Definitie 4.2 *Twee Boolese functies, f en g, in n veranderlijken zijn gelijk als en slechts als*

$$f(x_1, x_2, \ldots, x_n) = g(x_1, x_2, \ldots, x_n)$$

voor alle $(x_1, x_2, \ldots, x_n) \in B^n$.

M.a.w. twee Boolese functies zijn gelijk als ze voor dezelfde input steeds dezelfde output hebben.

Voorbeeld
Beschouw twee functies f en g met

$$\begin{aligned} f(x,y) &= \overline{(x+\overline{y})} \cdot \overline{x} \\ g(x,y) &= \overline{x+y}. \end{aligned}$$

Zowel f als g is een functie in twee variabelen. De beide functies zijn gelijk indien ze voor dezelfde input dezelfde output opleveren. Dit kan snel geverifieerd worden a.d.h.v. een outputtabel. We gebruiken één tabel voor de beide functies:

x	y	$\overline{x}$	$\overline{y}$	$x+\overline{y}$	$x+y$	$f(x,y)$	$g(x,y)$
0	0	1	1	1	0	1	1
0	1	1	0	0	1	0	0
1	0	0	1	1	1	0	0
1	1	0	0	1	1	0	0

De twee laatste kolommen uit de tabel geven aan dat de beide functies steeds voor eenzelfde input eenzelfde output genereren. Hieruit mogen we besluiten dat de Boolese functie f gelijk is aan de functie g.

4.2 Minimale en maximale Boolese uitdrukkingen

Uit het voorgaande blijkt dat twee functies in evenveel variabelen met een verschillend functievoorschrift toch hetzelfde resultaat kunnen opleveren. Dit betekent dat het functievoorschrift voor het bekomen van een bepaalde output niet uniek is.
Gelijke functies kunnen herleid worden naar eenzelfde *standaardvorm*.

We bespreken twee verschillende standaardvormen.
De eerste standaardvorm wordt de *disjunctieve normaalvorm* (DNV) van de functie genoemd. De tweede standaardvorm wordt de *conjunctieve normaalvorm* (CNV) genoemd. De conjunctieve normaalvorm is de duale van de disjunctieve normaalvorm.

Beide standaardvormen, zowel DNV als CNV, zijn uniek. Enerzijds betekent dit dat elke functie juist één DNV en één CNV heeft; anderzijds betekent dit dat twee gelijke functies eenzelfde DNV en eenzelfde CNV hebben.

De disjunctieve normaalvorm wordt opgebouwd als een som van *minimale uitdrukkingen*. De conjunctieve normaalvorm, die de duale is van de disjunctieve normaalvorm, is op zijn beurt een product van *maximale uitdrukkingen*.

Definitie 4.3 *Een Boolese uitdrukking in n variabelen, $x_1, x_2, \ldots, x_n$ is* ***minimaal*** *als ze het product is van n factoren en waarbij de k-de factor x_k of $\overline{x}_k$ is.*

Definitie 4.4 *Een Boolese uitdrukking in n variabelen, $x_1, x_2, \ldots, x_n$ is* ***maximaal*** *als ze de som is van n termen en waarbij de k-de term x_k of $\overline{x}_k$ is.*

Voorbeelden

- Minimale uitdrukkingen:

in één variabele x	in twee variabelen x en y	in drie variabelen x, y en z
		$x \cdot y \cdot z$
	$x \cdot y$	$x \cdot y \cdot \overline{z}$
x		$x \cdot \overline{y} \cdot z$
	$x \cdot \overline{y}$	$x \cdot \overline{y} \cdot \overline{z}$
		$\overline{x} \cdot y \cdot z$
	$\overline{x} \cdot y$	$\overline{x} \cdot y \cdot \overline{z}$
$\overline{x}$		$\overline{x} \cdot \overline{y} \cdot z$
	$\overline{x} \cdot \overline{y}$	$\overline{x} \cdot \overline{y} \cdot \overline{z}$

- Maximale uitdrukkingen:

in één variabele x	in twee variabelen x en y	in drie variabelen x, y en z
		$x + y + z$
	$x + y$	$x + y + \overline{z}$
x		$x + \overline{y} + z$
	$x + \overline{y}$	$x + \overline{y} + \overline{z}$
		$\overline{x} + y + z$
	$\overline{x} + y$	$\overline{x} + y + \overline{z}$
$\overline{x}$		$\overline{x} + \overline{y} + z$
	$\overline{x} + \overline{y}$	$\overline{x} + \overline{y} + \overline{z}$

4.3 De disjunctieve normaalvorm: DNV

Definitie 4.5 *De* ***disjunctieve normaalvorm (DNV)*** *van een Boolese uitdrukking in n variabelen is een som van minimale uitdrukkingen in n variabelen.*

Elke Boolese functie kan herleid worden tot haar DNV, die uniek is. Twee gelijke functies hebben steeds dezelfde DNV.

Voorbeelden
De volgende Boolese uitdrukkingen zijn genoteerd in hun disjunctieve normaalvorm:

- $f(x,y) = (x \cdot y) + (\overline{x} \cdot y) + (x \cdot \overline{y})$
- $f(x,y,z) = (\overline{x} \cdot y \cdot z) + (\overline{x} \cdot \overline{y} \cdot z) + (\overline{x} \cdot \overline{y} \cdot \overline{z})$.

Een Boolese functie herleiden tot haar DNV kan op verschillende manieren. We bekijken twee mogelijke methodes.

4.3.1 Methode 1: gebruik axioma's en eigenschappen

Gebruikmakend van de axioma's en eigenschappen van een Boole-algebra kunnen we de DNV afleiden uit de gegeven uitdrukking.

Voorbeelden

1. Stel $f : B^3 \to B : x \mapsto f(x,y,z) = y\overline{z} + (y+z)(\overline{z+\overline{x}})$.

$$\begin{array}{rcll}
f(x,y,z) & = & y\overline{z} + (y+z)(\overline{z+\overline{x}}) & \\
& = & y\overline{z} + (y+z)\overline{z}\,\overline{\overline{x}} & \text{(wet van de Morgan)} \\
& = & y\overline{z} + (y+z)\overline{z}x & \text{(involutie)} \\
& = & y\overline{z} + y\overline{z}x + z\overline{z}x & \text{(distributieve wet)} \\
& = & y\overline{z} + y\overline{z}x + 0x & \text{(complementwet)} \\
& = & y\overline{z} + y\overline{z}x & \text{(begrenzing, identiteitswet)} \\
& = & 1y\overline{z} + y\overline{z}x & \text{(identiteitswet)} \\
& = & (x+\overline{x})y\overline{z} + y\overline{z}x & \text{(complementwet)} \\
& = & xy\overline{z} + \overline{x}y\overline{z} + y\overline{z}x & \text{(distributieve wet)} \\
& = & xy\overline{z} + \overline{x}y\overline{z} & \text{(idempotentie, commutatieve wet)}
\end{array}$$

Besluit: $f(x,y,z) = (x \cdot y \cdot \overline{z}) + (\overline{x} \cdot y \cdot \overline{z})$ (DNV).

2. Stel $f : B^3 \to B : x \mapsto f(x,y,z) = (x \cdot \overline{y} + \overline{x} \cdot y) \cdot (x+z)$.

$$\begin{array}{ll}
f(x,y,z) & \\
= (x \cdot \overline{y} + \overline{x} \cdot y) \cdot (x+z) & \\
= x \cdot \overline{y} \cdot x + \overline{x} \cdot y \cdot x + x \cdot \overline{y} \cdot z + \overline{x} \cdot y \cdot z & \text{(distributieve wet)} \\
= x \cdot \overline{y} + 0 \cdot y + x \cdot \overline{y} \cdot z + \overline{x} \cdot y \cdot z & \text{(comm. wet, idempotentie, compl. wet)} \\
= x \cdot \overline{y} + x \cdot \overline{y} \cdot z + \overline{x} \cdot y \cdot z & \text{(begrenzing, identiteitswet)} \\
= x \cdot \overline{y} + \overline{x} \cdot y \cdot z & \text{(absorptie)} \\
= x \cdot \overline{y} \cdot 1 + \overline{x} \cdot y \cdot z & \text{(identiteitswet)} \\
= x \cdot \overline{y} \cdot (z + \overline{z}) + \overline{x} \cdot y \cdot z & \text{(complementwet)} \\
= x \cdot \overline{y} \cdot z + x \cdot \overline{y} \cdot \overline{z} + \overline{x} \cdot y \cdot z & \text{(distributieve wet)}
\end{array}$$

Besluit: $f(x,y,z) = (x \cdot \overline{y} \cdot z) + (x \cdot \overline{y} \cdot \overline{z}) + (\overline{x} \cdot y \cdot z)$ (DNV).

4.3.2 Methode 2: gebruik de uitvoertabel

Voor een Boolese functie in n variabelen $x_1, \ldots, x_n$, zijn 2^n mogelijkheden voor de input. Met elke inputcombinatie correspondeert één minimale uitdrukking.

input	corresponderende minimale uitdrukking
$(1,1,\ldots,1)$	$x_1 \cdot x_2 \cdot \ldots \cdot x_n$
$(0,1,\ldots,1)$	$\overline{x_1} \cdot x_2 \cdot \ldots \cdot x_n$
$\vdots$	$\vdots$
$(0,0,\ldots,0)$	$\overline{x_1} \cdot \overline{x_2} \cdot \ldots \cdot \overline{x_n}$

Indien de inputwaarde voor de variabele x_i $(i = 1, \ldots, n)$ 0 is dan komt in de corresponderende minimale term $\overline{x_i}$ voor, is de inputwaarde 1 dan komt in de corresponderende minimale term x_i voor.

Eigenschap 4.1 *Elke Boolese uitdrukking in n variabelen kan voorgesteld worden door de volgende som:*

$$f(x_1, x_2, \ldots, x_n) =$$
$$f(1,1,\ldots,1)x_1x_2\ldots x_n + f(0,1,\ldots,1)\overline{x}_1x_2\ldots x_n + \cdots + f(0,0,\ldots,0)\overline{x}_1\overline{x}_2\ldots\overline{x}_n.$$

Eigenschap 4.1 biedt een oplossingsmethode voor het bepalen van de DNV van een functie.

Voorbeelden

1. Stel $f : B^2 \to B : x \mapsto f(x,y) = x + \overline{x}y$.
   Gelet op eigenschap 4.1 kan $f(x,y)$ ook genoteerd worden als som van alle minimale termen in twee variabelen telkens vermenigvuldigd met hun corresponderende outputwaarde.

   $$f(x,y) = f(1,1) \cdot x \cdot y + f(1,0) \cdot x \cdot \overline{y} + f(0,1) \cdot \overline{x} \cdot y + f(0,0) \cdot \overline{x} \cdot \overline{y}$$

   Voor het bepalen van de outputwaarden gebruiken we een tabel:

x	y	$\overline{x}$	$\overline{x} \cdot y$	$f(x,y) = x + \overline{x}y$
0	0	1	0	0
0	1	1	1	1
1	0	0	0	1
1	1	0	0	1

   We substitueren de gevonden outputwaarden in de uitdrukking voor f. We vinden:

   $$\begin{aligned} & f(x,y) \\ &= f(1,1) \cdot x \cdot y + f(1,0) \cdot x \cdot \overline{y} + f(0,1) \cdot \overline{x} \cdot y + f(0,0) \cdot \overline{x} \cdot \overline{y} \\ &= 1 \cdot x \cdot y + 1 \cdot x \cdot \overline{y} + 1 \cdot \overline{x} \cdot y + 0 \cdot \overline{x} \cdot \overline{y} \\ &= x \cdot y + x \cdot \overline{y} + \overline{x} \cdot y + 0 && \text{(identiteitswet, begrenzing)} \\ &= x \cdot y + x \cdot \overline{y} + \overline{x} \cdot y. && \text{(identiteitswet)} \end{aligned}$$

   Dit resultaat is niets anders dan de DNV van de functie f.
   We analyseren even de gevonden DNV. Welke minimale termen zijn overgebleven in de DNV van f?
   Enkel de termen waarvoor de bijhorende functiewaarde 1 is, komen voor in de DNV van f. Alle andere minimale termen vallen weg gelet op de begrenzingseigenschap.

   Samengevat kunnen we stellen dat de DNV van een functie de som is van alle minimale termen die corresponderen met een outputwaarde gelijk aan 1.

2. Stel $f : B^3 \to B : x \mapsto x \cdot (\overline{y} + z)$
Uit het eerste voorbeeld hebben we geleerd dat het niet nodig is om de volledige som, waartoe het functievoorschrift kan herleid worden conform eigenschap 4.1, uit te schrijven. Uit de outputtabel kan rechtstreeks afgeleid worden welke minimale termen tot de DNV zullen behoren en welke niet. Enkel deze termen die corresponderen met een outputwaarde 1 behoren tot de DNV van f.
De outputtabel voor f is:

x	y	z	$\overline{y}$	$\overline{y}+z$	$f(x,y,z)$
0	0	0	1	1	0
0	0	1	1	1	0
0	1	0	0	0	0
0	1	1	0	1	0
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	0	0	0
1	1	1	0	1	1

Er zijn drie inputcombinaties die aanleiding geven tot een outputwaarde gelijk aan 1: $(1,0,0)$, $(1,0,1)$ en $(1,1,1)$.
De drie minimale termen die corresponderen met deze inputcombinaties zullen de enige minimale termen zijn die in de DNV van f voorkomen. Deze minimale termen zijn: $x \cdot \overline{y} \cdot \overline{z}$, $x \cdot \overline{y} \cdot z$ en $x \cdot y \cdot z$.
Alle minimale termen die corresponderen met een outputwaarde 0 vallen weg, gelet op de begrenzing.
Dit resulteert in de volgende DNV voor de functie f

$$f(x,y,z) = (x \cdot y \cdot z) + (x \cdot \overline{y} \cdot z) + (x \cdot \overline{y} \cdot \overline{z}).$$

4.4 De conjunctieve normaalvorm: CNV

Wanneer een functie wordt genoteerd in zijn DNV dan wordt deze als een som van minimale uitdrukkingen voorgesteld.
De conjunctieve normaalvorm (CNV) is de duale van de DNV. Dit betekent dat de CNV geen *som* maar een *product* zal zijn en niet van *minimale* maar wel van *maximale* uitdrukkingen.

Definitie 4.6 *De **conjunctieve normaalvorm (CNV)** van een uitdrukking in n variabelen is een product van maximale uitdrukkingen in n variabelen.*

Net als voor de DNV heeft elke functie juist één CNV en hebben gelijke functies dezelfde CNV.

Voorbeelden
De volgende Boolese uitdrukkingen zijn genoteerd in hun conjunctieve normaalvorm.

- $f(x,y) = (x+y) \cdot (\overline{x}+y)$
- $f(x,y,z) = (\overline{x}+y+z) \cdot (\overline{x}+\overline{y}+z) \cdot (\overline{x}+\overline{y}+\overline{z})$.

Een Boolese functie herleiden tot haar CNV kan, net zoals voor de DNV, op verschillende manieren. Daar de tweede methode, nl. werken met de outputtabel, vaak de voorkeur krijgt, zullen we ons voor de CNV beperken tot deze methode.

Methode: gebruik de uitvoertabel

De werkwijze, voor het bepalen van de DNV via de outputtabel, is een rechtstreeks gevolg van eigenschap 4.1.

De oplossingsmethode voor het bepalen van de CNV komt overeen met de duale werkwijze: dezelfde stappen worden doorlopen maar telkens de duale vorm ervan.

Methode DNV	Methode CNV
Stel de outputtabel op. Zoek alle outputwaarden gelijk aan **1**. Construeer de bijhorende **minimale term**: • als input x_i = **1** dan x_i in minimale term • als input x_i = **0** dan $\overline{x_i}$ in minimale term. Verbind alle minimale termen met een $+$.	Stel de outputtabel op. Zoek alle outputwaarden gelijk aan **0**. Construeer de bijhorende **maximale uitdrukking**: • als input x_i = **0** dan x_i in maximale uitdrukking • als input x_i = **1** dan $\overline{x_i}$ in maximale uitdrukking. Verbind alle maximale uitdrukkingen met een $\cdot$.

Voorbeeld

Hernemen we hetzelfde voorbeeld als voor DNV:

$$f : B^3 \to B : x \mapsto x \cdot (\overline{y}+z).$$

Outputtabel:

x	y	z	$\overline{y}$	$\overline{y}+z$	$f(x,y,z)$
0	0	0	1	1	0
0	0	1	1	1	0
0	1	0	0	0	0
0	1	1	0	1	0
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	0	0	0
1	1	1	0	1	1

Voor het opstellen van de CNV gaan we op zoek naar de outputwaarden 0 in de tabel. Er zijn vijf inputcombinaties die resulteren in output 0, nl.: $(0,0,0)$, $(0,0,1)$, $(0,1,0)$, $(0,1,1)$, $(1,1,0)$.
Voor elk van deze inputcombinaties moet de corresponderende maximale uitdrukking opgebouwd worden. Het product van al deze maximale uitdrukkingen vormt de CNV van f.

Noteren we de maximale uitdrukkingen in dezelfde volgorde als de opgesomde inputcombinaties dan ziet de CNV er als volgt uit:

$$f(x,y,z) = (x+y+z)\cdot(x+y+\overline{z})\cdot(x+\overline{y}+z)\cdot(x+\overline{y}+\overline{z})\cdot(\overline{x}+\overline{y}+z).$$

4.5 Vereenvoudigen van Boolese uitdrukkingen

De DNV en CNV van een functie zijn uniek. Gelijke functies hebben eenzelfde DNV (resp. CNV).
Zowel de CNV als de DNV zijn over het algemeen niet de eenvoudigste voorstelling van de corresponderende Boolese functie.

Een Boolese functie vereenvoudigen kan o.a. door te steunen op de axioma's en eigenschappen van een Boole-algebra. Welke axioma's en eigenschappen er moeten toegepast worden om de uitdrukking te vereenvoudigen, daar zijn geen vaste regels voor. Het beste resultaat vindt men via *trial and error*.

In het volgende hoofdstuk behandelen we een andere, snellere, manier voor het vereenvoudigen van Boolese uitdrukkingen, nl. *Karnaugh-diagrammen*.

4.6 Oefeningen

1. Bepaal de DNV en CNV van $f(x,y,z)$ als:

   a) $f(1,1,1) = f(1,0,1) = f(1,1,0) = f(0,0,0) = 1$. Alle andere waarden van f zijn 0.

b) $f(0,0,0) = f(0,1,0) = f(0,1,1) = 1$. Alle andere waarden van f zijn 0.

2. Bepaal de DNV van $f(x,y,z,u)$ als:

   a) $f(1,1,1,1) = f(1,0,0,1) = f(1,0,1,0) = 1$. Alle andere waarden van f zijn 0.

   b) $f(1,0,1,0) = f(0,0,0,0) = f(0,1,0,1) = f(1,1,1,1) = 1$. Alle andere waarden van f zijn 0.

3. Geef de DNV (op twee verschillende manieren) en CNV van:

   a) $f(x,y,z) = \overline{x+y+z}$

   b) $f(x,y,z) = ((\overline{x}+y)+\overline{z}) \cdot \overline{z} \cdot x$

   c) $f(x,y,z) = (x \cdot (\overline{y}+z)) + \overline{z}$

4. Vereenvoudig:

   a) $(x+y) \cdot (\overline{y} \cdot \overline{z}) + (x \cdot \overline{y})$

   b) $x \cdot \overline{y} + (\overline{x}+y) \cdot y$

   c) $((x+y)+x \cdot z) \cdot (x+y \cdot z)$

   d) $(x+x \cdot y+x \cdot y \cdot z) \cdot (x \cdot y+x \cdot z) \cdot (y+x \cdot y \cdot z)$

4.7 Extra oefeningen

1. Bereken de gevraagde waarden van de volgende Boole-formules:

a)	$f(x) = \overline{x}$	$f(0), f(1)$
b)	$f(x,y) = (x+\overline{y}) \cdot (\overline{x}+\overline{y})$	$f(1,1), f(0,1)$
c)	$f(x,y,z) = (\overline{x \cdot y \cdot z}) \cdot (x+\overline{y}+z)$	$f(1,0,1), f(0,0,1)$
d)	$f(x,y,z,u) = x \cdot y \cdot \overline{u} + \overline{y} \cdot z \cdot u + x \cdot \overline{z} \cdot u$	$f(1,1,1,1), f(1,1,0,0)$

2. Geef alle minimale termen in vier veranderlijken x,y,z,u.

3. Geef alle maximale termen in vier veranderlijken x,y,z,u.

4. Stel de DNV en CNV op van de functies waarvoor de gegeven waarden gelijk zijn aan 1. Alle niet vermelde waarden zijn gelijk aan 0.

   a) $f(1) = 1$

   b) $f(1,1) = f(0,0) = 1$

   c) $f(1,1,1) = f(0,1,1) = f(1,0,0) = f(0,0,0) = 1$

   d) $f(0,1,1) = f(1,0,0) = f(0,1,0) = 1$

5. Bepaal de DNV van de volgende Boolese functies.

a) $f(x, y) = x + \overline{y} \cdot (x + \overline{y}) \cdot (\overline{x} + y)$

b) $f(x, y, z) = (x \cdot y + \overline{y} \cdot z) \cdot (\overline{x} + y \cdot z) + (\overline{x \cdot y \cdot z})$

c) $f(x, y, z, u) = (x \cdot y + z \cdot u) \cdot (x \cdot \overline{z} + \overline{y} \cdot u) \cdot (y \cdot z + x \cdot u)$

6. Vereenvoudig de gegeven Boolese uitdrukkingen door gebruik te maken van eigenschappen en axioma's.

   a) $(x + y) \cdot (\overline{x + y}) + x$

   b) $\overline{(x + y) \cdot (\overline{x} + \overline{y})}$

   c) $(x + y + \overline{x} \cdot \overline{z}) \cdot (\overline{x + y + z})$

   d) $\overline{(x + y) \cdot (\overline{x} + z) \cdot (y + z)}$

   e) $(\overline{x \cdot y + \overline{y} \cdot z}) \cdot (\overline{\overline{y} \cdot \overline{z} + \overline{x} \cdot y}) \cdot (\overline{\overline{x} + \overline{y} \cdot z})$

   f) $\overline{x} \cdot (y \cdot z \cdot u + \overline{z} + \overline{u}) + (\overline{x \cdot (\overline{y} + u)})$

4.8 Oplossingen

1.

a)	1	0
b)	0	0
c)	1	1
d)	0	1

2. $x \cdot y \cdot z \cdot u$, $x \cdot y \cdot z \cdot \overline{u}$, $x \cdot y \cdot \overline{z} \cdot u$, $x \cdot y \cdot \overline{z} \cdot \overline{u}$, $x \cdot \overline{y} \cdot z \cdot u$, $x \cdot \overline{y} \cdot z \cdot \overline{u}$, $x \cdot \overline{y} \cdot \overline{z} \cdot u$, $x \cdot \overline{y} \cdot \overline{z} \cdot \overline{u}$, $\overline{x} \cdot y \cdot z \cdot u$, $\overline{x} \cdot y \cdot z \cdot \overline{u}$, $\overline{x} \cdot y \cdot \overline{z} \cdot u$, $\overline{x} \cdot y \cdot \overline{z} \cdot \overline{u}$, $\overline{x} \cdot \overline{y} \cdot z \cdot u$, $\overline{x} \cdot \overline{y} \cdot z \cdot \overline{u}$, $\overline{x} \cdot \overline{y} \cdot \overline{z} \cdot u$, $\overline{x} \cdot \overline{y} \cdot \overline{z} \cdot \overline{u}$.

3. $x + y + z + u$, $x + y + z + \overline{u}$, $x + y + \overline{z} + u$, $x + y + \overline{z} + \overline{u}$,
   $x + \overline{y} + z + u$, $x + \overline{y} + z + \overline{u}$, $x + \overline{y} + \overline{z} + u$, $x + \overline{y} + \overline{z} + \overline{u}$,
   $\overline{x} + y + z + u$, $\overline{x} + y + z + \overline{u}$, $\overline{x} + y + \overline{z} + u$, $\overline{x} + y + \overline{z} + \overline{u}$,
   $\overline{x} + \overline{y} + z + u$, $\overline{x} + \overline{y} + z + \overline{u}$, $\overline{x} + \overline{y} + \overline{z} + u$, $\overline{x} + \overline{y} + \overline{z} + \overline{u}$.

4. DNV:

   a) $f(x) = x$

   b) $f(x, y) = (x \cdot y) + (\overline{x} \cdot \overline{y})$

   c) $f(x, y, z) = (\overline{x} \cdot \overline{y} \cdot \overline{z}) + (\overline{x} \cdot y \cdot z) + (x \cdot \overline{y} \cdot \overline{z}) + (x \cdot y \cdot z)$

   d) $f(x, y, z) = (\overline{x} \cdot y \cdot \overline{z}) + (\overline{x} \cdot y \cdot z) + (x \cdot \overline{y} \cdot \overline{z})$

   CNV:

   a) $f(x) = x$

   b) $f(x, y) = (x + \overline{y}) \cdot (\overline{x} + y)$

   c) $f(x, y, z) = (x + y + \overline{z}) \cdot (x + \overline{y} + z) \cdot (\overline{x} + y + \overline{z}) \cdot (\overline{x} + \overline{y} + z)$

d) $f(x,y,z) = (x+y+z)\cdot(x+y+\overline{z})\cdot(\overline{x}+y+\overline{z})\cdot(\overline{x}+\overline{y}+z)\cdot(\overline{x}+\overline{y}+\overline{z})$

5. a) $f(x,y) = x\cdot y + x\cdot\overline{y} + \overline{x}\cdot\overline{y}$

   b) $f(x,y,z) = x\cdot y\cdot z + \overline{x}\cdot y\cdot z + x\cdot\overline{y}\cdot z + x\cdot y\cdot\overline{z} + x\cdot\overline{y}\cdot\overline{z} + \overline{x}\cdot y\cdot\overline{z} + \overline{x}\cdot\overline{y}\cdot z + \overline{x}\cdot\overline{y}\cdot\overline{z}$

   c) $f(x,y,z,u) = x\cdot y\cdot\overline{z}\cdot u + x\cdot\overline{y}\cdot z\cdot u$

6. a) x

   b) $x\cdot y + \overline{x}\cdot\overline{y}$

   c) $\overline{x}\cdot\overline{y}\cdot\overline{z}$

   d) $\overline{x}\cdot\overline{y} + x\cdot\overline{z}$

   e) 0

   f) $\overline{x} + y\cdot\overline{u}$

HOOFDSTUK 5

Veitch-Karnaugh diagrammen

In het vorige hoofdstuk zijn twee verschillende oplossingsmethodes besproken voor het bepalen van de DNV en CNV van een Boolese functie. Een eerste methode maakte gebruik van axioma's en eigenschappen. De tweede methode maakte gebruik van de tabel.

Voor het vereenvoudigen van een functie bespreken we opnieuw twee manieren. De eerste manier is via eigenschappen en axioma's zoals reeds geïllustreerd werd in het voorgaande hoofdstuk. De tweede manier maakt gebruik van een tabel.
De outputtabel die gebruikt wordt voor het vereenvoudigen van een Boolese functie is echter niet de tabel zoals we ze tot nu toe opgesteld hebben, maar wel het *Veitch-Karnaugh diagram* (KD).

Een Veitch-Karnaugh diagram is eveneens een outputtabel. Dit betekent dat voor alle mogelijke inputcombinaties de corresponderende output terug te vinden is in de tabel. Alleen wordt deze informatie op een andere manier geordend in een KD dan in een 'gewone' outputtabel.

Wanneer we verwijzen naar een Karnaugh diagram zullen we steeds de volledige naam gebruiken. Indien we refereren naar een tabel zoals we tot nu toe steeds gebruikt hebben dan zullen we deze benoemen als outputtabel.

Een Veitch-Karnaugh diagram (KD) van een Boolese functie f in n variabelen is een tabel waarin de 2^n outputwaarden van de functie voorkomen. Hoe een dergelijke tabel moet opgesteld worden, wordt in de volgende paragraaf besproken.

5.1 KD van een Boolese uitdrukking

In een outputtabel worden alle mogelijke inputcombinaties onder elkaar genoteerd en naast de input wordt de corresponderende output geconstrueerd.

Een KD wordt opgesteld als een rooster waarbij de input wordt uitgezet langs de rijen

en de kolommen.
De mogelijke inputcombinaties worden voorgesteld d.m.v. de corresponderende minimale termen.

In het rooster zelf worden enkel de outputwaarden van de functie genoteerd.
Waar welke output komt te staan is afhankelijk van de ordening van de minimale termen aan de rand van de tabel.

Deze minimale termen mogen echter niet zomaar willekeurig geordend worden. De ordening moet aan een aantal opgelegde regels voldoen.

Vorm van een KD voor een functie in n variabelen

- Orden alle minimale termen, in n variabelen, zodanig dat twee aan elkaar grenzende termen slechts in één variabele van elkaar verschillen.
- Plaats een 1 bij alle minimale termen die voorkomen in de DNV van de functie. De minimale termen die voorkomen in de DNV zijn precies deze die corresponderen met een outputwaarde 1. Deze minimale termen geven dus exact aan waar de 1-en moeten ingevuld worden in het KD. Op de overige posities komt een 0 te staan.

We starten met het eenvoudigste geval.

5.1.1 KD voor een functie in 1 veranderlijke

Voor een willekeurige Boolese functie $f(x)$ in één veranderlijke x zijn er twee inputcombinaties mogelijk $x = 1$ en $x = 0$.
De minimale term die correspondeert met $x = 1$ is x zelf en de minimale term die correspondeert met input 0 is $\overline{x}$.
Deze twee minimale termen worden uitgezet aan de rand van de tabel, in de tabel zelf komen dan de corresponderende outputwaarden te staan.

Algemeen

Voor een willekeurige functie $f(x)$ ziet het KD er als volgt uit:

$\overline{x}$	x
$f(0)$	$f(1)$

Voorbeeld

Stel $f : B \to B : x \mapsto f(x) = x + \overline{x}$.

De corresponderende outputtabel is:

x	$\overline{x}$	$f(x) = x + \overline{x}$
0	1	1
1	0	1

In deze tabel zien we dat $f(0) = 1$ en $f(1) = 1$.
Gebruiken we deze informatie voor het opstellen van het KD horend bij deze functie f, dan vinden we voor het KD:

$\overline{x}$	x
1	1

5.1.2 KD voor een functie in 2 veranderlijken

Algemeen

Gelet op het feit dat alle minimale termen, die aan elkaar grenzen, maar op één positie mogen verschillen van elkaar is de volgende ordening een goede ordening voor een KD van een functie in twee variabelen x en y.

	$\overline{y}$	y
$\overline{x}$	$f(0,0)$	$f(0,1)$
x	$f(1,0)$	$f(1,1)$

Voorbeeld

Stel $f : (x,y) \mapsto x + y$.

Outputtabel:

x	y	$x+y$
0	0	0
0	1	1
1	0	1
1	1	1

KD:

	$\overline{y}$	y
$\overline{x}$	0	1
x	1	1

5.1.3 KD voor een functie in 3 veranderlijken

Algemeen

Voor een functie in 3 variabelen x, y en z voldoet de volgende ordening aan de regels van een KD.

	$\overline{z}$	z
$\overline{x} \cdot \overline{y}$	$f(0,0,0)$	$f(0,0,1)$
$\overline{x} \cdot y$	$f(0,1,0)$	$f(0,1,1)$
$x \cdot y$	$f(1,1,0)$	$f(1,1,1)$
$x \cdot \overline{y}$	$f(1,0,0)$	$f(1,0,1)$

Waar ook we staan in de tabel: doen we één stap naar boven, links, rechts of onder, de corresponderende input zal telkens maar op één positie gewijzigd zijn.
Hierbij moeten we rekening houden met het feit dat het KD beschouwd wordt als een cilinder die je kan toeplooien langs beide zijden. Dit betekent: de bovenste rij grenst aan de onderste rij, de laatste kolom grenst aan de eerste kolom.

Dus elk element in de tabel grenst aan vier andere elementen: een element onder, boven, links en rechts; en de corresponderende input van de aan elkaar grenzende functiewaarden is slechts op één positie verschillend.

Voorbeeld

Beschouw de functie $f : (x, y, z) \mapsto (x \cdot y \cdot z) + (x \cdot \overline{y} \cdot z) + (\overline{x} \cdot \overline{y} \cdot \overline{z})$.

Het gegeven functievoorschrift van deze functie f komt overeen met haar DNV.
We weten dat voor elke minimale term uit de DNV de corresponderende input een outputwaarde 1 heeft. Alle overige inputcombinaties hebben output 0.

Dit betekent dat het functievoorschrift alle informatie bevat die nodig is om het KD in te vullen. We weten precies waar de outputwaarden 1 zullen staan: enkel bij de drie minimale termen die voorkomen in de DNV.

Het bijhorende KD is:

	$\overline{z}$	z
$\overline{x} \cdot \overline{y}$	1	0
$\overline{x} \cdot y$	0	0
$x \cdot y$	0	1
$x \cdot \overline{y}$	0	1

5.1.4 KD voor een functie in 4 veranderlijken

Algemeen

Voor een functie in vier variabelen x, y, z en u ziet de opbouw van het KD er als volgt uit:

	$\overline{z} \cdot \overline{u}$	$\overline{z} \cdot u$	$z \cdot u$	$z \cdot \overline{u}$
$\overline{x} \cdot \overline{y}$	$f(0,0,0,0)$	$f(0,0,0,1)$	$f(0,0,1,1)$	$f(0,0,1,0)$
$\overline{x} \cdot y$	$f(0,1,0,0)$	$f(0,1,0,1)$	$f(0,1,1,1)$	$f(0,1,1,0)$
$x \cdot y$	$f(1,1,0,0)$	$f(1,1,0,1)$	$f(1,1,1,1)$	$f(1,1,1,0)$
$x \cdot \overline{y}$	$f(1,0,0,0)$	$f(1,0,0,1)$	$f(1,0,1,1)$	$f(1,0,1,0)$

Voorbeeld

Beschouw de functie

$$f : (x, y, z, u) \mapsto (x \cdot y \cdot z \cdot u) + (\overline{x} \cdot \overline{y} \cdot \overline{z}) + (x \cdot \overline{y}).$$

Het functievoorschrift van de functie is een som maar niet de DNV van de functie. Niet alle termen zijn minimale termen in 4 variabelen.
Indien we de DNV kennen dan kan het KD snel ingevuld worden. Uit de opgegeven som is het echter heel eenvoudig de DNV af te leiden.
Indien voor de gegeven uitdrukking de DNV zou afgeleid worden m.b.v. eigenschappen en axioma's zouden we elke term waarin een variabele ontbreekt moeten uitbreiden tot een minimale term van vier variabelen. Om dit te bereiken moet de term vermenigvuldigd worden met 1 en deze 1 wordt dan uitgeschreven als de som van het ontbrekende element met zijn complement. Na toepassing van de distributieve wet, komen de gezochte minimale termen tevoorschijn.
In dit specifieke geval betekent dit: de eerste term in het functievoorschrift is reeds een minimale term in 4 variabelen; deze moet niet meer aangepast worden en zal zeker in de DNV voorkomen. In de tweede term ontbreekt de variabele u, indien de zonet beschreven stappen gevolgd worden, zullen we twee minimale termen bekomen, nl. $\overline{x} \cdot \overline{y} \cdot \overline{z} \cdot u$ en $\overline{x} \cdot \overline{y} \cdot \overline{z} \cdot \overline{u}$, die eveneens tot de DNV behoren. In de laatste term van het opgegeven functievoorschrift ontbreken twee variabelen nl. z en u. Als we deze term uitbreiden tot een minimale term in vier variabelen dan zal dit resulteren in 4 minimale termen in vier variabelen die allemaal zullen voorkomen in de DNV van de functie, nl. $x \cdot \overline{y} \cdot z \cdot u$, $x \cdot \overline{y} \cdot z \cdot \overline{u}$, $x \cdot \overline{y} \cdot \overline{z} \cdot u$ en $x \cdot \overline{y} \cdot \overline{z} \cdot \overline{u}$.

Dus

$$f(x, y, z, u) = \\ (x \cdot y \cdot z \cdot u) + (\overline{x} \cdot \overline{y} \cdot \overline{z} \cdot u) + (\overline{x} \cdot \overline{y} \cdot \overline{z} \cdot \overline{u}) + (x \cdot \overline{y} \cdot z \cdot u) + (x \cdot \overline{y} \cdot z \cdot \overline{u}) + (x \cdot \overline{y} \cdot \overline{z} \cdot u) + (x \cdot \overline{y} \cdot \overline{z} \cdot \overline{u}).$$

Het KD opstellen is nu zeer eenvoudig. De gevonden minimale termen, die tot de DNV behoren, moeten corresponderen met een outputwaarde 1.

Het corresponderende KD is:

	$\overline{z} \cdot \overline{u}$	$\overline{z} \cdot u$	$z \cdot u$	$z \cdot \overline{u}$
$\overline{x} \cdot \overline{y}$	1	1	0	0
$\overline{x} \cdot y$	0	0	0	0
$x \cdot y$	0	0	1	0
$x \cdot \overline{y}$	1	1	1	1

5.1.5 KD voor een functie in 5 veranderlijken

Algemeen

Bekijken we als laatste voorbeeld de opbouw van een KD voor een functie in vijf variabelen x, y, z, u en v.

	$\overline{u} \cdot \overline{v}$	$\overline{u} \cdot v$	$u \cdot v$	$u \cdot \overline{v}$
$\overline{x} \cdot \overline{y} \cdot \overline{z}$	$f(0,0,0,0,0)$	$\dots$		
$\overline{x} \cdot \overline{y} \cdot z$	$\vdots$			
$\overline{x} \cdot y \cdot z$				
$\overline{x} \cdot y \cdot \overline{z}$				
$x \cdot y \cdot \overline{z}$				
$x \cdot y \cdot z$				
$x \cdot \overline{y} \cdot z$				
$x \cdot \overline{y} \cdot \overline{z}$				

Voorbeeld

Beschouw de functie

$$f(x,y,z,u,v) = (x \cdot y \cdot z \cdot u) + (\overline{z} \cdot u \cdot v) + (x \cdot v).$$

De outputtabel opstellen voor deze functie om hieruit het KD af te leiden is tijdrovend aangezien het gaat over een functie met 5 inputwaarden. Voor 5 veranderlijken zijn er $2^5 = 32$ inputcombinaties mogelijk.

Anderzijds hebben we in het voorgaande voorbeeld uitgelegd hoe we uit een gegeven som de DNV kunnen afleiden. Elke term in de som moet uitgebreid worden tot een minimale term. Alle mogelijke uitbreidingen van de term zullen in de DNV voorkomen. Alle minimale termen uit de DNV corresponderen met een 1 in het KD.

In de eerste term $x \cdot y \cdot z \cdot u$ ontbreekt één variabele, nl. v. Er zijn twee manieren om deze term uit te breiden tot een minimale term: v toevoegen of $\overline{v}$ toevoegen. Beiden minimale termen zullen in de DNV voorkomen. Het is niet nodig de DNV volledig uit te schrijven, we kunnen dit onmiddellijk aanduiden in het KD door op de overeenkomstige plaatsen een outputwaarde 1 te noteren.
Evenzo voor de tweede term. In deze term ontbreken twee variabelen, nl. x en y. Er zijn vier manieren (2 voor x en 2 voor y, in totaal 2×2 mogelijkheden) om deze term uit te breiden tot een minimale term, plaatsen we opnieuw bij alle vier deze termen een 1 in het KD.
In de laatste term van het gegeven functievoorschrift ontbreken drie variabelen dit resulteert in $2^3 = 8$ mogelijkheden om deze term uit te breiden tot een minimale term. Al deze acht minimale termen zullen tot de DNV behoren, ook voor deze termen moet een outputwaarde 1 ingevuld worden in het KD.
Indien op een bepaalde positie al een 1 staat dan laten we deze gewoon staan. Elke term kan maar één keer voorkomen in de DNV (cfr. idempotentie).

Dit resulteert in het volgende KD:

	$\overline{u} \cdot \overline{v}$	$\overline{u} \cdot v$	$u \cdot v$	$u \cdot \overline{v}$
$\overline{x} \cdot \overline{y} \cdot \overline{z}$	0	0	1	0
$\overline{x} \cdot \overline{y} \cdot z$	0	0	0	0
$\overline{x} \cdot y \cdot z$	0	0	0	0
$\overline{x} \cdot y \cdot \overline{z}$	0	0	1	0
$x \cdot y \cdot \overline{z}$	0	1	1	0
$x \cdot y \cdot z$	0	1	1	1
$x \cdot \overline{y} \cdot z$	0	1	1	0
$x \cdot \overline{y} \cdot \overline{z}$	0	1	1	0

5.2 Vereenvoudigde vorm

Het KD moet ons helpen een Boolese functie te vereenvoudigen.

De outputtabel laat toe om snel DNV en CNV af te lezen zonder enig gebruik van axioma's of eigenschappen. Evenzo laat het KD toe een functie te vereenvoudigen zonder enig gebruik van axioma's of eigenschappen.

5.2.1 Algoritme

1. Stel het KD op.

2. Vorm met de 1-en uit het KD rechthoeken.
   Het aantal 1-en in de rechthoek moet een macht van twee zijn $(1, 2, 4, 8, \ldots)$. Koppel terug naar de minimale termen die corresponderen met de 1-en uit de rechthoek. Indien in deze minimale termen een variabele samen met haar complement voorkomt moet het aantal minimale termen waarin deze variabele voorkomt gelijk zijn aan het aantal minimale termen met het complement van de variabele.
   Maak de rechthoek zo groot mogelijk.
   Elke outputwaarde 1 moet in minstens één rechthoek voorkomen. De rechthoeken mogen overlappen maar mogen zeker geen outputwaarde 0 bevatten. Hoe minder rechthoeken hoe eenvoudiger het voorschrift zal worden.

3. Met elke gevonden rechthoek correspondeert een Boolese uitdrukking.
   Bekijk alle minimale termen die corresponderen met de enen uit de rechthoek. Onthoud uit deze minimale termen enkel de variabelen die zonder hun complement voorkomen. Het product van alle overgebleven variabelen is de uitdrukking corresponderend met deze rechthoek.

4. Maak een som van alle gevonden uitdrukkingen. Deze som is de vereenvoudigde vorm van de gegeven uitdrukking.

5.2.2 Voorbeeld

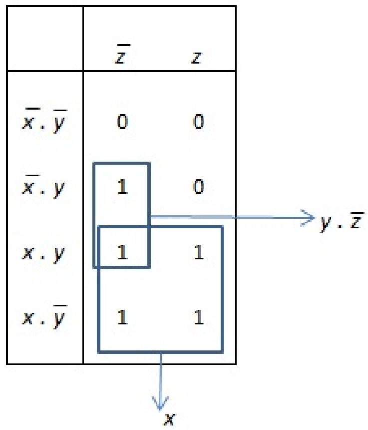
Beschouw de functie

$$f(x,y,z) = x + (y \cdot (x + \overline{z})).$$

Na uitwerken van de haakjes (distributieve wet) vinden we

$$f(x,y,z) = x + (y \cdot x) + (y \cdot \overline{z}).$$

Het functievoorschrift van f is omgezet naar een som. Uit deze som kan het KD corresponderend met de functie f afgeleid worden. Het KD is opgenomen in figuur 5.1.



Figuur 5.1:

In dezelfde figuur zijn, met alle 1-en die voorkomen in het diagram, rechthoeken opgebouwd zodat elke 1 in minstens één rechthoek voorkomt.
Voor elke rechthoek is vervolgens de bijhorende uitdrukking opgesteld: beschouw alle minimale uitdrukkingen die voorkomen in de rechthoek, elke variabele die maar op één manier voorkomt in deze uitdrukkingen wordt in de bijhorende uitdrukking opgenomen.
De som van al deze uitdrukkingen is de vereenvoudigde vorm van de oorspronkelijk gegeven functie.

In dit specifiek geval betekent dit:

$$f(x,y,z) = x + (y \cdot \overline{z}).$$

5.3 Oefeningen

Maak gebruik van Karnaugh-diagrammen om volgende Boolese functies te vereenvoudigen

1. $f(x,y,z) = \overline{x \cdot y \cdot z + \overline{x} \cdot \overline{y} \cdot z + x \cdot \overline{y} \cdot z + \overline{x} \cdot y \cdot z}$
2. $f(x,y,z,u) = y + \overline{z} \cdot u + \overline{y} \cdot \overline{z} \cdot \overline{u} + x \cdot y \cdot z \cdot u$
3. $f(x,y,z,u,v) = x \cdot \overline{z} \cdot \overline{v} + z \cdot u \cdot \overline{v} + z \cdot \overline{u} \cdot \overline{v} + x \cdot y \cdot z \cdot v$
4. $f(x,y,z,u) = x \cdot u \cdot (y \cdot z + \overline{y} \cdot \overline{z}) + z \cdot u \cdot (x \cdot \overline{y} + \overline{x} \cdot y) + \overline{z} \cdot (x \cdot y \cdot u + \overline{x} \cdot \overline{y} \cdot \overline{u})$
5. $f(x,y,z,u,v) = u \cdot \overline{v} + x \cdot \overline{v} + x \cdot u \cdot v + z \cdot \overline{u} \cdot \overline{v} + y \cdot \overline{z} \cdot \overline{v} + \overline{x} \cdot \overline{y} \cdot \overline{z} \cdot \overline{u} \cdot \overline{v}$
6. $f(x,y,z) = (x + \overline{y}) \cdot (x + z) \cdot (\overline{x} + \overline{z}) \cdot (\overline{y} + \overline{z})$

5.4 Extra oefeningen

1. Teken het KD van de gegeven Boolese uitdrukkingen in 3 veranderlijken.
   a) $x \cdot \overline{y} \cdot z + \overline{x} \cdot y \cdot z + \overline{x} \cdot \overline{y} \cdot \overline{z}$
   b) $\overline{y} \cdot z + x \cdot \overline{z}$
   c) $x + y + z$
2. Teken het KD van de gegeven Boolese uitdrukkingen in 4 veranderlijken.
   a) $x \cdot y \cdot \overline{z} \cdot u + \overline{x} \cdot y \cdot z \cdot \overline{u} + \overline{x} \cdot \overline{y} \cdot z \cdot \overline{u}$
   b) $(x \cdot y + \overline{x} \cdot \overline{y}) \cdot (z \cdot u + \overline{z} \cdot \overline{u})$
   c) $x + y + z + u$
3. Teken het KD van de gegeven Boolese uitdrukkingen in 5 of 6 veranderlijken.
   a) $x \cdot y \cdot \overline{z} \cdot u \cdot v + \overline{x} \cdot z \cdot \overline{v}$
   b) $y \cdot z \cdot (x \cdot \overline{u} \cdot \overline{v} \cdot w + \overline{x} \cdot u \cdot v \cdot \overline{w})$
   c) $v \cdot w + z \cdot u + x \cdot y \cdot \overline{z} \cdot u \cdot \overline{v} \cdot \overline{w}$
4. Noteer voor elk KD de corresponderende Boolese uitdrukking. Leid uit het KD de eenvoudigste uitdrukking af.

   a)

	z	$\overline{z}$
$x \cdot y$		
$x \cdot \overline{y}$	1	
$\overline{x} \cdot \overline{y}$	1	1
$\overline{x} \cdot y$		

b)

	$\overline{z} \cdot \overline{u}$	$\overline{z} \cdot u$	$z \cdot u$	$z \cdot \overline{u}$
$\overline{x} \cdot \overline{y}$			1	
$\overline{x} \cdot y$		1	1	1
$x \cdot y$		1	1	1
$x \cdot \overline{y}$			1	

c)

	$\overline{z} \cdot \overline{u}$	$\overline{z} \cdot u$	$z \cdot u$	$z \cdot \overline{u}$
$\overline{x} \cdot \overline{y}$		1	1	1
$\overline{x} \cdot y$		1	1	1
$x \cdot y$		1	1	1
$x \cdot \overline{y}$		1	1	1

d)

	$\overline{z} \cdot \overline{u}$	$\overline{z} \cdot u$	$z \cdot u$	$z \cdot \overline{u}$
$\overline{x} \cdot \overline{y}$				1
$\overline{x} \cdot y$				1
$x \cdot y$	1			1
$x \cdot \overline{y}$	1			1

e)

	$\overline{u} \cdot \overline{v}$	$\overline{u} \cdot v$	$u \cdot v$	$u \cdot \overline{v}$
$\overline{x} \cdot \overline{y} \cdot \overline{z}$				
$\overline{x} \cdot \overline{y} \cdot z$	1			1
$\overline{x} \cdot y \cdot z$				
$\overline{x} \cdot y \cdot \overline{z}$				
$x \cdot y \cdot \overline{z}$		1	1	
$x \cdot y \cdot z$		1	1	1
$x \cdot \overline{y} \cdot z$	1		1	1
$x \cdot \overline{y} \cdot \overline{z}$				

f)

	$\overline{u}\cdot\overline{v}\cdot\overline{w}$	$\overline{u}\cdot\overline{v}\cdot w$	$\overline{u}\cdot v\cdot w$	$\overline{u}\cdot v\cdot\overline{w}$	$u\cdot v\cdot\overline{w}$	$u\cdot v\cdot w$	$u\cdot\overline{v}\cdot w$	$u\cdot\overline{v}\cdot\overline{w}$
$\overline{x}\cdot\overline{y}\cdot\overline{z}$								
$\overline{x}\cdot\overline{y}\cdot z$	1	1	1	1	1	1	1	1
$\overline{x}\cdot y\cdot z$	1	1	1	1	1	1	1	1
$\overline{x}\cdot y\cdot\overline{z}$					1			1
$x\cdot y\cdot\overline{z}$				1	1			1
$x\cdot y\cdot z$				1	1			1
$x\cdot\overline{y}\cdot z$				1				
$x\cdot\overline{y}\cdot\overline{z}$				1				

5. Vereenvoudig de gegeven Boolese uitdrukkingen met de methode van Veitch-Karnaugh:

   a) $f(x,y) = \overline{x}\cdot y + x\cdot\overline{y} + (\overline{x+y})$
   b) $f(x,y,z) = x\cdot z + y\cdot\overline{z} + \overline{y}\cdot z + \overline{x}\cdot y\cdot z + \overline{x}\cdot\overline{y}\cdot\overline{z}$
   c) $f(x,y,z) = x\cdot(y+z)\cdot(\overline{y}+z) + x\cdot y\cdot\overline{z} + x\cdot\overline{y}\cdot z$
   d) $f(x,y,z,u) = \overline{z}\cdot\overline{u} + x\cdot y\cdot z\cdot u + x\cdot\overline{y}\cdot z\cdot u + \overline{x}\cdot y\cdot z\cdot u + \overline{x}\cdot\overline{y}\cdot z\cdot u$
   e) $f(x,y,z,u,v) = \overline{x}\cdot\overline{y} + \overline{x}\cdot\overline{z}\cdot(\overline{u}+v+y) + y\cdot\overline{z}\cdot\overline{v} + x\cdot y\cdot\overline{z}\cdot(u+\overline{v}) + x\cdot y\cdot\overline{z}\cdot\overline{u}\cdot v$
   f) $f(x,y,z,u,v,w) = \overline{x\cdot z\cdot u\cdot v + \overline{y}\cdot z\cdot u\cdot w + \overline{x}\cdot z\cdot u\cdot v + y\cdot z\cdot u\cdot w + z\cdot u\cdot\overline{v}\cdot\overline{w}}$

5.5 Oplossingen

1. a)

	$\overline{z}$	z
$\overline{x}\cdot\overline{y}$	1	
$\overline{x}\cdot y$		1
$x\cdot y$		
$x\cdot\overline{y}$		1

   b)

	$\overline{z}$	z
$\overline{x}\cdot\overline{y}$		1
$\overline{x}\cdot y$		
$x\cdot y$	1	
$x\cdot\overline{y}$	1	1

   c)

	$\overline{z}$	z
$\overline{x}\cdot\overline{y}$		1
$\overline{x}\cdot y$	1	1
$x\cdot y$	1	1
$x\cdot\overline{y}$	1	1

2. a)

	$\overline{z} \cdot \overline{u}$	$\overline{z} \cdot u$	$z \cdot u$	$z \cdot \overline{u}$
$\overline{x} \cdot \overline{y}$				1
$\overline{x} \cdot y$				1
$x \cdot y$		1		
$x \cdot \overline{y}$				

b)

	$\overline{z} \cdot \overline{u}$	$\overline{z} \cdot u$	$z \cdot u$	$z \cdot \overline{u}$
$\overline{x} \cdot \overline{y}$	1		1	
$\overline{x} \cdot y$				
$x \cdot y$	1		1	
$x \cdot \overline{y}$				

c)

	$\overline{z} \cdot \overline{u}$	$\overline{z} \cdot u$	$z \cdot u$	$z \cdot \overline{u}$
$\overline{x} \cdot \overline{y}$		1	1	1
$\overline{x} \cdot y$	1	1	1	1
$x \cdot y$	1	1	1	1
$x \cdot \overline{y}$	1	1	1	1

3. a)

	$\overline{u} \cdot \overline{v}$	$\overline{u} \cdot v$	$u \cdot v$	$u \cdot \overline{v}$
$\overline{x} \cdot \overline{y} \cdot \overline{z}$				
$\overline{x} \cdot \overline{y} \cdot z$	1			1
$\overline{x} \cdot y \cdot z$	1			1
$\overline{x} \cdot y \cdot \overline{z}$				
$x \cdot y \cdot \overline{z}$			1	
$x \cdot y \cdot z$				
$x \cdot \overline{y} \cdot z$				
$x \cdot \overline{y} \cdot \overline{z}$				

b)

	$\overline{u} \cdot \overline{v} \cdot \overline{w}$	$\overline{u} \cdot \overline{v} \cdot w$	$\overline{u} \cdot v \cdot w$	$\overline{u} \cdot v \cdot \overline{w}$	$u \cdot v \cdot \overline{w}$	$u \cdot v \cdot w$	$u \cdot \overline{v} \cdot w$	$u \cdot \overline{v} \cdot \overline{w}$
$\overline{x} \cdot \overline{y} \cdot \overline{z}$								
$\overline{x} \cdot \overline{y} \cdot z$								
$\overline{x} \cdot y \cdot z$					1			
$\overline{x} \cdot y \cdot \overline{z}$								
$x \cdot y \cdot \overline{z}$								
$x \cdot y \cdot z$		1						
$x \cdot \overline{y} \cdot z$								
$x \cdot \overline{y} \cdot \overline{z}$								

c)

	$\overline{u} \cdot \overline{v} \cdot \overline{w}$	$\overline{u} \cdot \overline{v} \cdot w$	$\overline{u} \cdot v \cdot w$	$\overline{u} \cdot v \cdot \overline{w}$	$u \cdot v \cdot \overline{w}$	$u \cdot v \cdot w$	$u \cdot \overline{v} \cdot w$	$u \cdot \overline{v} \cdot \overline{w}$
$\overline{x} \cdot \overline{y} \cdot \overline{z}$			1			1		
$\overline{x} \cdot \overline{y} \cdot z$			1		1	1	1	1
$\overline{x} \cdot y \cdot z$			1		1	1	1	1
$\overline{x} \cdot y \cdot \overline{z}$			1			1		
$x \cdot y \cdot \overline{z}$			1			1		1
$x \cdot y \cdot z$			1		1	1	1	1
$x \cdot \overline{y} \cdot z$			1		1	1	1	1
$x \cdot \overline{y} \cdot \overline{z}$			1			1		

4. a) $\overline{x} \cdot \overline{y} + \overline{y} \cdot z$
   b) $y \cdot z + y \cdot u + z \cdot u$
   c) $z + u$
   d) $x \cdot \overline{u} + z \cdot \overline{u}$
   e) $\overline{x} \cdot \overline{y} \cdot z \cdot \overline{v} + x \cdot y \cdot v + x \cdot z \cdot u + x \cdot \overline{y} \cdot z \cdot \overline{v} = x \cdot y \cdot v + x \cdot z \cdot u + \overline{y} \cdot z \cdot \overline{v}$
   f) $\overline{x} \cdot z + y \cdot u \cdot v \cdot \overline{w} + x \cdot \overline{u} \cdot v \cdot \overline{w} + y \cdot u \cdot \overline{v} \cdot \overline{w} = \overline{x} \cdot z + y \cdot u \cdot \overline{w} + x \cdot \overline{u} \cdot v \cdot \overline{w}$

5. a) $\overline{x} + \overline{y}$
   b) $\overline{x} + y + z$
   c) $x \cdot y + x \cdot z$
   d) $z \cdot u + \overline{z} \cdot \overline{u}$
   e) $\overline{x} \cdot \overline{y} + y \cdot \overline{z}$
   f) $\overline{y} \cdot \overline{z} + y \cdot \overline{z} + \overline{u} = \overline{z} + \overline{u}$

HOOFDSTUK **6**

Codeertheorie

Wanneer mensen met elkaar in gesprek zijn treedt er regelmatig miscommunicatie op. De gesprekspartner begrijpt bijvoorbeeld niet alle woorden die de andere persoon gebruikt, of delen van het gesprek worden niet goed verstaan. Ook bij de communicatie tussen computers, of tussen computers en randapparatuur, kunnen zulke problemen optreden.
Beschouw bijvoorbeeld het afspelen van een cd of dvd. Tijdens het afspelen vertaalt een laser de minuscule putjes op de disk naar enen en nullen, die door de apparatuur in muziek worden omgezet. Door stofjes of krassen op de disk, gebeurt het regelmatig dat een 1 per ongeluk voor een 0 wordt aangezien, of andersom. Ook bij het versturen van enen en nullen over het internet, of via satellieten, treden er regelmatig zulke verwisselingen op.
Dus wanneer informatie op één of andere wijze verstuurd wordt door een zender naar een ontvanger, dan kan deze informatie onderweg aangetast worden door fouten. De ontvanger moet dan maar de originele boodschap zien te achterhalen. De discipline die zich hiermee bezighoudt, noemen we *Codeertheorie*. We starten met een inleiding en bestuderen dan zowel enkele foutdetecterende codes als foutcorrigerende codes.

6.1 Inleiding

Wanneer informatie digitaal wordt opgeslagen, verstuurd en verwerkt, gebeurt dit onder de vorm van binaire code. Dan wordt deze informatie vertaald naar een rij van nullen en enen.

Voorbeeld De string ’SOS’ wordt omgezet, gecodeerd naar de binaire code 1011011100

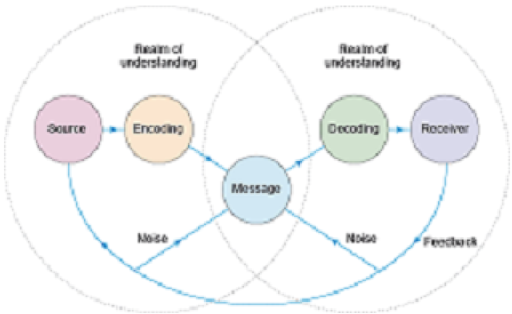
Bij het uitwisselen van informatie over een (communicatie)kanaal, kan het voorkomen dat niet alle bits correct worden doorgegeven. Ergens, bij het versturen of ontvangen, treden één of meerdere fouten op.

Voorbeeld

1. Het verstuurde bericht is: 10110**1**1100
2. Het ontvangen bericht is: 10110**0**1100

Het kanaal waarover verstuurd wordt, kan een telefoonlijn, een satellietverbinding of een computerverbinding zijn.

Algemeen gezien, werkt een communicatiesysteem als volgt:



1. De verzender maakt een bericht.
2. Het bericht wordt gecodeerd.
3. Het gecodeerd bericht wordt verstuurd over het communicatiekanaal.
4. Het gecodeerd bericht wordt ontvangen.
5. Het gecodeerd bericht wordt gedecodeerd naar het originele bericht.
6. Het bericht wordt afgeleverd aan de ontvanger.

Door het versturen van een bericht over een kanaal met **ruis** of dus een communicatiekanaal dat onderhevig is aan storingen, kan het ontvangen bericht verschillend zijn van het verzonden bericht:

Er kunnen verschillende soorten ruis onderscheiden worden.

Natuurlijke ruis: Dit is een verstoring van de communicatie door natuurlijke fenomenen zoals bliksem, temperatuurschommelingen.

Toevallige ruis: Hier spelen onvoorspelbare vormen van communicatiebeschadiging zoals magnetische fluctuaties, slijtage van een communicatiedrager mee en zorgen voor fouten.

Opzettelijke ruis: In dit geval wordt de communicatie doelbewust verstoord, dit gebeurt bij het inrichten van stoorsignalen.

We hebben dus nood aan een systeem om **bitfouten**, op zijn minst, te detecteren en liefst ook te verbeteren.

We maken hiervoor gebruik van de codeertheorie, waarbij er extra (overtollige) informatie wordt toegevoegd aan het (oorspronkelijke) bericht, zodat we kunnen achterhalen of het ontvangen bericht correct is of niet.

We maken onderscheid tussen foutdetecterende en foutverbeterende codes.

Om codes te vergelijken maken we gebruik van de performantie van de code:

Definitie 6.1 ***Performantie van de code:*** *de verhouding van de hoeveelheid bitfouten die kunnen ontdekt worden t.o.v. de hoeveelheid extra informatie die toegevoegd moet worden aan de te versturen (of op te slagen) bitreeks.*

6.2 Foutdetecterende codes

Deze codes ontdekken dat er fouten zijn, als het ontvangen bericht verschillend is van het verzonden bericht. Deze codes corrigeren de fouten **niet**. Dit is ook niet altijd nodig.

Bij videocodering algoritmes zal een fout een spikkeltje of een blokje in één beeld veroorzaken, dit is niet zo erg.

We bekijken enkele soorten foutdetecterende codes:

6.2.1 Herhaling

Algoritme De oorspronkelijke bitreeks wordt n keer herhaald.

Voorbeeld:
n = 2
De oorspronkelijke data is : 1011101
We versturen : 10111011011101
We ontvangen : 10111011001101
We zien dat niet 2 (=n) keer hetzelfde ontvangen werd, dus er is een fout opgetreden.

Af te spreken:

1. de lengte van de oorspronkelijke bitreeks

2. n (het aantal herhalingen).

Performantie: zéér laag
Er moet zeer veel extra informatie verzonden worden.

6.2.2 Pariteit

Er zijn twee vormen, namelijk oneven pariteit en even pariteit.

Algoritme We voegen één bit (= de pariteitsbit) toe, zodanig dat de binaire code een oneven aantal 1-bits (logische enen) heeft (dit noemen we oneven pariteit), of zodanig dat de binaire code een even aantal 1-bits (logische enen) heeft (dit noemen we even pariteit).

De waarde van de pariteitsbit is afhankelijk van de gekozen pariteit:

Oneven pariteit:
Als het aantal 1-bits van de binaire code oneven is, dan is de pariteitsbit 0.
Als het aantal 1-bits van de binaire code even is, dan is de pariteitsbit 1.

Het totaal aantal 1-bits moet steeds oneven zijn.

Even pariteit:
Als het aantal 1-bits van de binaire code oneven is, dan is de pariteitsbit 1.
Als het aantal 1-bits van de binaire code even is, dan is de pariteitsbit 0.

Het totaal aantal 1-bits moet steeds even zijn.

Voorbeeld: oneven pariteit
De oorspronkelijke data is 1011101, de oorspronkelijke bitreeks heeft 5 logische enen.
we versturen 1011101**0** want we hebben 5 eentjes, dit is een oneven aantal enen. Aangezien we met oneven pariteit werken is de pariteitsbit 0, zodat het aantal eentjes oneven is/blijft.
We ontvangen 10101010
De ontvangen code heeft maar 4 eentjes, aangezien we met oneven pariteit werken moet het aantal enen oneven zijn, er zit dus een FOUT in de ontvangen data!

Af te spreken:

1. van de oorspronkelijke (te versturen) bitreeks
2. oneven of even pariteit

3. de plaats van de pariteitsbit (we zetten de pariteitsbit voor- of achteraan)

Performantie: laag
alle oneven aantallen bitfouten worden herkend
alle even aantallen bitfouten worden NIET herkend
Positief is wel dat er weinig extra bits worden verzonden.

6.2.3 Controlegetal - rest als controlegetal

Algoritme de rest (modulo) bij deling door een priemgetal vormt een controlegetal.

Met behulp van modulo rekenen kun je een coderingssysteem maken. Hierbij werken we bijvoorbeeld met modulo 13. Dit betekent dat we werken met het 13-tallig stelsel met de cijfers: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 (= A), 11 (= B), 12 (= C). In dit voorbeeld maken we codewoorden met vijf codecijfers en twee pariteitscijfers. Een code heeft dus de volgende opbouw: [c0 c1 c2 c3 c4 p5 p6], waarbij c5 en c6 de pariteitscijfers zijn. Een parititeitscijfer is in dit voorbeeld een 'getal' van 0 t/m 12. Tip: schrijf bij de opgaven de getallen A, B en C om naar de decimale notatie, dus 10, 11 en 12, want dit rekent gemakkelijker. De waarde van de pariteitscijfers wordt bepaald door de volgende twee formules:

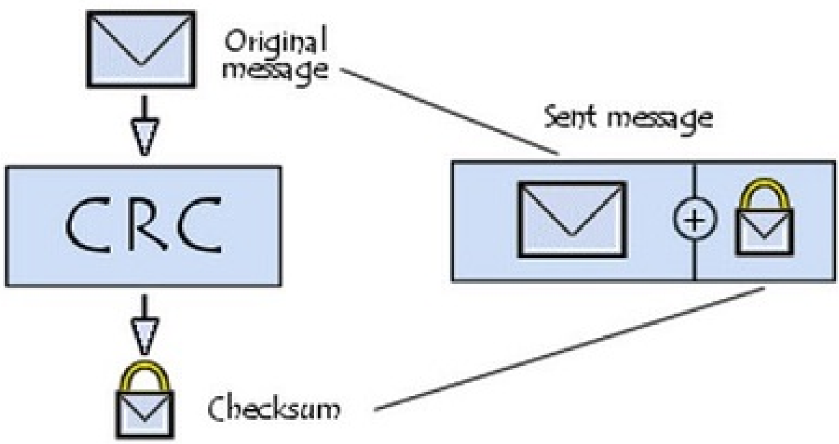
$$\begin{cases} c_0 + c_1 + c_2 + c_3 + c_4 + p_5 + p_6 \equiv \mathbf{0}(mod\ 13) \\ c_1 + 2 \cdot c_2 + 3 \cdot c_3 + 4 \cdot c_4 + 5 \cdot p_5 + 6 \cdot p_6 \equiv \mathbf{0}(mod\ 13) \end{cases}$$

Voorbeeld
Een bankrekeningnummer : BE05 3631 9038 54**75**
De laatste 2 cijfers vormen het controlegetal
Het controlegetal wordt bepaald door 3631903854 te delen door het grootste priemgetal, dat kleiner is dan 100 , namelijk 97. De rest van deze deling vormt hierbij het controlegetal en wordt achteraan toegevoegd. Hier is het controlegetal 75.
Indien de rest 0 is, wordt 97 als controlegetal gebruikt.
Let op: BE05 is de landcode (BE: in dit geval België) gevolgd door een op een andere manier berekend controlegetal (05) voor het IBAN-formaat.

Uiteraard kan dit dan omgezet worden naar binaire waarden.

6.2.4 Controlegetal - Cyclic Redundancy Check

Een cyclic redundancy check (CRC) is een foutdetecterende code die dikwijls gebruikt wordt in digitale netwerken en opslagmedia om bitfouten te detecteren. Blokken data die deze systemen binnenkomen, krijgen een korte controlewaarde of 'checksum' gebaseerd op de rest bij een 'deling met rest' op de data. Bij het binnenhalen of lezen van de data wordt de 'deling met rest' opnieuw uitgevoerd, als daar dezelfde checksum uitkomt, is de data zeer waarschijnlijk correct.



CRC heeft haar naam te danken aan het feit dat het algoritme gebaseerd is op cyclische codes (Cyclic). De checksum is redundant wat wil zeggen dat ze geen extra info bevat. Tenslotte controleert (check) CRC de data.
CRC's worden zeer vaak toegepast omdat ze makkelijk te implementeren zijn in binaire hardware, gemakkelijk wiskundig te analyseren zijn, en goed zijn in het detecteren van veelvoorkomende fouten geproduceerd door ruis in een transmissiekanaal. CRC is uitgevonden door W. Wesley Peterson in 1961.

Voordelen:

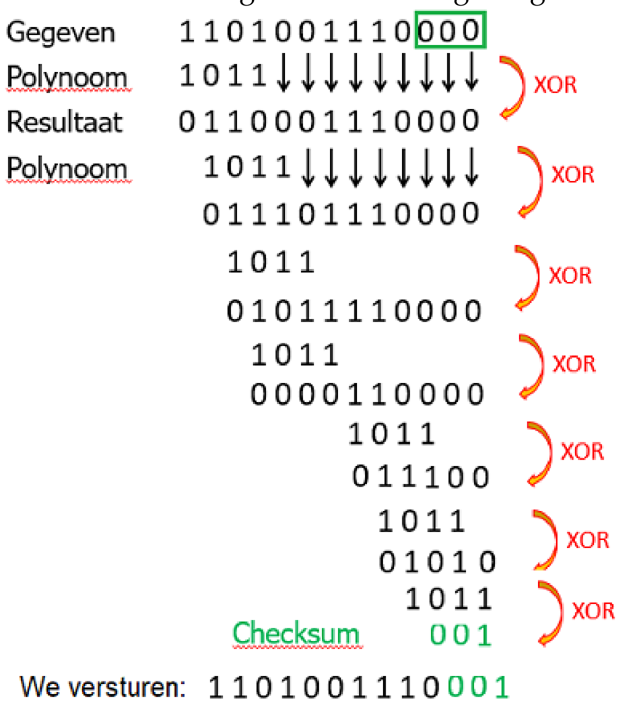
1. Veel krachtiger dan pariteit
2. Eenvoudig in hardware te implementeren Daarom wordt CRC zeer veel gebruikt in computerhardware: harde schijven, Ethernet, IP, TCP, USB, PCI-e, Wifi, ...

Definitie Als de binaire waarde van de data gedeeld wordt door de CRC-polynoom, ontstaat er een checksum. Deze checksum wordt achter de data geplaatst. De data en de checksum kunnen nu samen opgeslagen of verstuurd worden.
Als later de data gelezen of ontvangen wordt, zal er weer een CRC-deling plaatsvinden. De deling deelt het binaire getal met daarachter de checksum door dezelfde CRC-polynoom. De deling zou geen rest mogen opleveren. Als er een fout plaatsgevonden heeft bij het opslaan of verzenden van de data, zal de rest meestal ongelijk zijn aan 0. De deling kan ook enkel op het binaire getal plaatsvinden; als de bekomen restwaarde gelijk is aan de checksum, is er geen fout gedetecteerd in de data.
De toepassingen die gebruikmaken van CRC zullen ervoor zorgen dat de data opnieuw opgeslagen of verzonden wordt tot er geen fouten meer gedetecteerd worden.

Algoritme

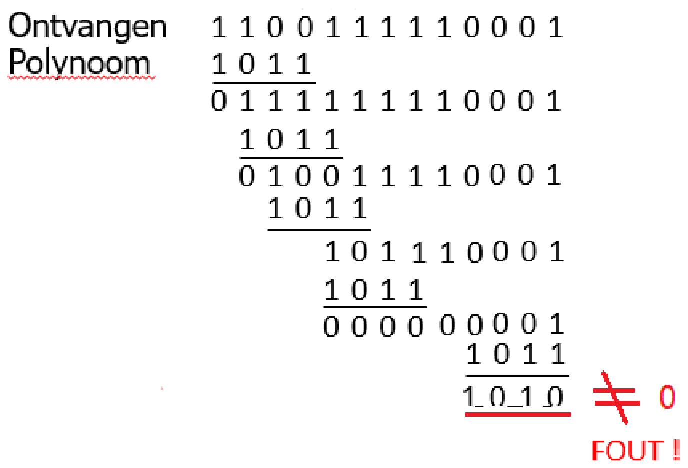
1. Schrijf achter de bitreeks (rechts) een aantal nullen, dit aantal is 1 minder dan de lengte van het polynoom. Dit zal ook de lengte zijn van de checksum
2. Schrijf de polynoom links onder deze bit-reeks.
3. Als de bit boven de meest linkse bit van de polynoom nul is gebeurt er niets en gaan we naar stap 5.
4. Als de bit boven de meest linkse bit van de polynoom 1 is wordt een bitsgewijze exclusieve OF uitgevoerd.
5. De polynoom wordt opnieuw onder het bekomen resultaat geschreven, één bit meer naar rechts en terug naar stap 2.
6. Stop wanneer het bekomen resultaat 1 bit korter is dan het polynoom.
7. Vervang de nullen door de bekomen checksum.

Voorbeeld: We willen de bitreeks 1101001110 versturen, hiervoor gebruiken we de CRC methode met als polynoom 1011.
We zullen dus 3 nullen moeten toevoegen aan de bitreeks (1 minder dan de lengte van de polynoom). daara starten we met de bitsgewijze exclusieve OF te berekenen.
Op het einde vervangen we de 3 toegevoegde nullen door de berekende checksum.



Resultaat: De te versturen bitreeks is 1101001110001 (de oorspronkelijke bitreeks gevolgd door de checksum)

Voorbeeld: We ontvangen een bitreeks 1100111110001 die gecodeerd is met de CRC methode die als polynoom 1011 heeft gebruikt. We moeten controleren of de ontvangen bitreeks juist is, zodat we de verstuurde bitreeks kennen.
We doen opnieuw de deling en controleren of de rest gelijk is aan nul.



Dus: op de bestemming wordt de CRC-deling opnieuw uitgevoerd en indien de rest niet 0 is, is er één fout of zijn er meerdere fouten opgetreden.

Af te spreken:

1. De lengte van de oorspronkelijke bitreeks
2. De gebruikte/te gebruiken polynoom
3. De plaats waar we de checksum toevoegen: voor- of achteraan

Performantie: goed
n-bit CRC zal in elke bitreeks een 'single error burst' (aaneengesloten bitreeks) van n bit kunnen detecteren
Er worden weinig extra bits verzonden: één minder dan de lengte van de polynoom

6.2.5 Oefeningen

Oefening 1 Bereken de CRC checksum van

1. data: 1101101 met polynoom 10011
2. data: 111010010 met polynoom 10011

3. data: 10111001011 met polynoom 100101
4. data: 110001110 met polynoom 110101
5. data: 11001010011 met polynoom 110101

Oefening 2 Je ontvangt onderstaande bitreeksen van een zender die een CRC checksum achter de originele gegevens meestuurt. Geef aan of de ontvangen bitreeksen correct of fout zijn.

1. data: 11001001101010 met polynoom 100101
2. data: 1011011010100011 met polynoom 101011
3. data: 11001110100110000 met polynoom 10111
4. data: 1001101100001100 met polynoom 10111
5. data: 1000111000110000 met polynoom 10111

Oplossing oefeningen

Oplossing oefening 1 Bereken de CRC checksum van

1. data: 1101101 met polynoom 10011 :
2. data: 111010010 met polynoom 10011 :
3. data: 10111001011 met polynoom 100101 :
4. data: 110001110 met polynoom 110101 :
5. data: 11001010011 met polynoom 110101 :

Oplossing oefening 2 e ontvangt onderstaande bitreeksen van een zender die een CRC checksum achter de originele gegevens meestuurt. Geef aan of de ontvangen bitreeksen correct of fout zijn.

1. data: 11001001101010 met polynoom 100101 :
2. data: 1011011010100011 met polynoom 101011 :
3. data: 11001110100110000 met polynoom 10111 :
4. data: 1001101100001100 met polynoom 10111 :
5. data: 1000111000110000 met polynoom 10111 :

6.2.6 Foutverbeterende codes

Deze codes ontdekken dat er fouten zijn, als het ontvangen bericht verschillend is van het verzonden bericht. Deze codes **verbeteren** ook het ontvangen bericht, zodat het oorspronkelijk bericht gedecodeerd wordt. De voorwaarde is wel dat er niet te veel fouten zijn opgetreden.

We maken onderscheid tussen:

Backward Error Correction Bij fouten worden de gegevens opnieuw opgevraagd aan de verzender.

Er is wel een beperking: enkel wanneer vertraging of latency voor de toepassing niet belangrijk is of wanneer het opnieuw versturen zéér snel kan, is dit bruikbaar. **Bijvoorbeeld** bij TCP, Ethernet, . . .

Forward Error Correction Algoritmen die in staat zijn om de correcte bitreeks te reconstrueren uit de vervormde bitreeks. Een voorbeeld hiervan is de Hammingcode

Hamming code

In de telecommunicatie is een Hamming-code een foutcorrigerende code, genoemd naar de uitvinder, Richard Hamming. Hamming-codes zijn lineaire codes, en zij kunnen 1 of 2 bitfouten detecteren, of 1 bitfout corrigeren.

Algoritme (aan de hand van een voorbeeld) Beschouw de te verzenden binaire code of bitreeks **1111000010101110**. Deze databits worden genummerd van 1 tot 16 en aangeduid als d_1 tot d_{16}. *Let op:* de bitnummering begint vanaf 1! Niet vanaf 0.

1	1	1	1	0	0	0	0	1	0	1	0	1	1	1	0
d1	d2	d3	d4	d5	d6	d7	d8	d9	d10	d11	d12	d13	d14	d15	d16

De Hammingcode voegt op specifieke plaatsen pariteitsbits, als controlebits, toe tussen de databits. De pariteitsbits worden aangeduid als p_1, p_2, p_3, p_4, p_5. De posities van deze pariteitsbits in de gecodeerde bitreeks worden bepaald door de machten van 2 en wel als volgt:

- p_1 op positie $2^0 = 1$
- p_2 op positie $2^1 = 2$
- p_3 op positie $2^2 = 4$
- p_4 op positie $2^3 = 8$

- p_5 op positie $2^4 = 16$

Het aantal gebruikte pariteitsbits is afhankelijk van de lengte van de code. We voegen de pariteitsbits op de juiste posities nu toe tussen de databits en krijgen volgende structuur:

		1		1	1	1		0	0	0	0	1	0	1		0	1	1	1	0
p1	p2	d1	p3	d2	d3	d4	p4	d5	d6	d7	d8	d9	d10	d11	p5	d12	d13	d14	d15	d16

p_1 staat dus op positie 1 van de gecodeerde bitreeks, p_2 op de 2de positie, enz.
We eindigen met p_5 omdat p_6 op positie 32 zou moeten staan, wat buiten het bereik valt.
Vervolgens wordt de waarde bepaald van de pariteitsbits, p_1 tot en met p_5. Hiervoor gebruikt de Hammingcodering altijd even pariteit!

Om dit te bepalen wordt er volgend schema opgesteld:

			1		1	1	1		0	0	0	0	1	0	1		0	1	1	1	0
	p1	p2	d1	p3	d2	d3	d4	p4	d5	d6	d7	d8	d9	d10	d11	p5	d12	d13	d14	d15	d16
p1 (1)	x		x		x		x		x		x		x		x		x		x		x
p2 (2)		x	x			x	x			x	x			x	x			x	x		
p3 (4)				x	x	x	x					x	x	x	x					x	x
p4 (8)								x	x	x	x	x	x	x	x						
p5 (16)																x	x	x	x	x	x

Schema:

- De structuur van de Hammingcode met de databits reeds ingevuld, staat bovenaan.
- Links staat een kolom met de gebruikte pariteitsbits in de Hammingcode met tussen haakjes de positie van deze bits in de code. Dit getal wordt ook gebruikt om aan te duiden welke bits van belang zijn per rij. Deze zijn aangeduid in het schema met een kruisje:

  Voorbeeld:

- Om de waarde van pariteitsbit p1 te bepalen, starten we op positie p1, we duiden 1 bit aan wisselen dan per bit. We selecteren dus één niet en daarna één wel.
- Om p_2 te bepalen, vertrekken we vanaf p_2, we selecteren 2 bits wel, dan 2 bits niet, enz.
- Dit doen we voor elk van de pariteitsbits ...

Om de juiste waarde van de pariteitsbits te bepalen, worden eerst per lijn de databits gekopieerd op de posities aangeduid door de kruisjes:

Vervolgens onderzoeken we per lijn de pariteit, die even moet zijn. Dus we tellen het aantal eentjes per lijn en als dit even is, wordt de pariteitsbit op die lijn 0, anders wordt deze 1.

																						pariteit
			1		1	1	1		0	0	0	0	1	0	1		0	1	1	1	0	
	p1	p2	d1	p3	d2	d3	d4	p4	d5	d6	d7	d8	d9	d10	d11	p5	d12	d13	d14	d15	d16	
p1 (1)	~~?~~0		1		1		1		0		0		1		1		0		1		0	6 = even => p1 = 0
p2 (2)		~~?~~0	1			1	1			0	0			0	1			1	1			6 = even => p2 = 0
p3 (4)				~~?~~0	1	1	1					0	1	0	1					1	0	6 = even => p3 = 0
p4 (8)								~~?~~0	0	0	0	0	1	0	1							2 = even => p4 = 0
p5 (16)																~~?~~1	0	1	1	1	0	3 = oneven => p5 = 1

De bekomen waardes van de pariteitsbits worden tussen de databits geplaatst, op de juiste positie:

	0	**0**	1	**0**	1	1	1	**0**	0	0	0	0	1	0	1	**1**	0	1	1	1	0
	p1	p2	d1	p3	d2	d3	d4	p4	d5	d6	d7	d8	d9	d10	d11	p5	d12	d13	d14	d15	d16
P1 (1)	**0**		1		1		1		0		0		1		1		0		1		0
P2 (2)		**0**	1			1	1			0	0			0	1			1	1		
P3 (4)				**0**	1	1	1					0	1	0	1					1	0
P4 (8)								**0**	0	0	0	0	1	0	1						
P5 (16)																**1**	0	1	1	1	0

De te verzenden Hammingcode van de (oorspronkelijke) bitreeks is dus
001**0**111**0**0000101**1**01110

Correctie van een fout:
Gegeven is de ontvangen Hammingcode van een bitreeks : 001001100000101101110.
Aan de hand van het schema, achterhalen we waar de fout zich bevindt.

	0	0	1	0	0	1	1	0	0	0	0	0	1	0	1	1	0	1	1	1	0	
	p1	p2	d1	p3	d2	d3	d4	p4	d5	d6	d7	d8	d9	d10	d11	p5	d12	d13	d14	d15	d16	Pariteit
p1 (1)	**0**		1		0		1		0		0		1		1		0		1		0	5 = oneven = fout !
p2 (2)		**0**	1			1	1			0	0			0	1			1	1			6 = even = correct
p3 (4)				**0**	0	1	1					0	1	0	1					1	0	5 = oneven = fout !
p4 (8)								**0**	0	0	0	0	1	0	1							2 = even = correct
p5 (16)																**1**	0	1	1	1	0	4 = even = correct

We noteren de posities van de pariteitsbits met een oneven aantal eentjes in hun rij en tellen deze op.
In het voorbeeld: p_1 en p_3 of dus 1 + 4 = 5 dus de 5e bit in de Hammingcode is verkeerd (in het rood aangeduid).
De verbeterde Hammingcode is 0010**1**1100000101101110
Het gedecodeerd bericht of de oorspronkelijke binaire code wordt bekomen door het verwijderen van de pariteitsbits en is dus 1111000010101110

Af te spreken:

1. De lengte van de oorspronkelijke bitreeks

Performantie: Niet zo goed

- De Hammingcode was het eerste algoritme voor Forward Error Correction.
- Er zijn ondertussen performantere algoritmes ontwikkeld:
  - Veterbi algoritme (GSM, Wifi)
  - Turbo codes (G3 netwerken)
  - LDPC - Low Density Parity Check (DVB-S2, Ethernet 10GBase-T)

6.2.7 Oefeningen

Oefening 1 Codeer volgende gegevens met de Hamming code

- 1100111
- 0001100
- 1000111

Oefening 2 Geef de oorspronkelijke data van ′10001100100′

Oefening 3 Geef de oorspronkelijke data van ′01100111011′

Oefening 4 Je ontvangt onderstaande gegevens die met de Hamming code gecodeerd werden. Corrigeer eventuele fouten en geef de oorspronkelijke data:

- 10010010011
- 11010100110
- 00000000100
- 11110100111

Oefening 5
Gegeven: Een ontvanger krijgt de volgende Hammingcode binnen

0 0 0 0 0 1 0 1 1 1 1 1

Gevraagd:

- Omcirkel in bovenstaande code de pariteitsbits of controlebits.
- Maak een kruisje over de foutieve bit. Indien er geen foute bit is noteer dan ′geen′: . . . .
- Geef de oorspronkelijke binaire code: . . . .

Oplossing oefeningen

HOOFDSTUK 7

Inleiding tot de Analyse

In dit deel van de cursus analyseren we reële functies. Deze reële functies worden opgebouwd in het veld $\mathbb{R}$.

We herhalen eerst een aantal notaties en belangrijke rekenregels die geldig zijn in $\mathbb{R}$.

7.1 Het veld $\mathbb{R}, +, \cdot$

Voor de verzameling der reële getallen, $\mathbb{R}$, samen met de optelling (+) en de vermenigvuldiging ($\cdot$) gedefinieerd op deze verzameling, gelden de volgende eigenschappen:

Stel $x, y, z \in \mathbb{R}$, dan geldt:

- $\mathbb{R}$ is gesloten voor $+$ en $\cdot$: $x + y \in \mathbb{R}$ en $x \cdot y \in \mathbb{R}$.
- commutatief: $x + y = y + x$ en $x \cdot y = y \cdot x$.
- associatief: $(x + y) + z = x + (y + z)$ en $(x \cdot y) \cdot z = x \cdot (y \cdot z)$.
- distributief: $x \cdot (y + z) = x \cdot y + x \cdot z$.
- neutraal element: $x + 0 = x$ en $x \cdot 1 = x$.
- invers element voor +: voor elke $x \in \mathbb{R}$ bestaat er een element $-x \in \mathbb{R}$ zodat $x + (-x) = 0$.
- invers element voor $\cdot$: voor elke $x \in \mathbb{R}_0$ bestaat er een element x^{-1} zodanig dat $x \cdot x^{-1} = 1$.

Hieruit mogen we besluiten dat $\mathbb{R}, +, \cdot$ een veld is.

Voorbeelden Stel $a, b \in \mathbb{R}$.

- $2a + 3a = (2+3)a = 5a$ (distributiviteit)
- $2(3a) = (2 \cdot 3)a = 6a$ (associativiteit)
- Gelet op de eigenschappen van een veld geldt:

$$\begin{aligned}
2(\frac{1}{2}a + b) - (a + \frac{1}{2}b) &= 2 \cdot \frac{1}{2} \cdot a + 2 \cdot b - a - \frac{1}{2} \cdot b \\
&= 2 \cdot 2^{-1} \cdot a + 2 \cdot b - a - \frac{1}{2} \cdot b \\
&= 1 \cdot a + 2 \cdot b - a - \frac{1}{2} \cdot b \\
&= (a - a) + (2 - \frac{1}{2}) \cdot b \\
&= \frac{4-1}{2} \cdot b \\
&= \frac{3}{2} \cdot b.
\end{aligned}$$

7.2 Machten

Definitie 7.1

- *Stel $a \in \mathbb{R}_0$ dan geldt voor alle $n \in \mathbb{N}_0$*

$$\begin{aligned}
a^0 &= 1 \\
a^1 &= a \\
a^n &= a \cdot a^{n-1} \quad (n \geq 2) \\
a^{-1} &= \tfrac{1}{a} \\
a^{-n} &= \tfrac{1}{a^n}.
\end{aligned}$$

- *Stel $a \in \mathbb{R}_0^+$ dan geldt voor alle $p \in \mathbb{N}$ en $q \in \mathbb{N}_0$*

$$\begin{aligned}
a^{\frac{1}{q}} &= \sqrt[q]{a} = w \Leftrightarrow w^q = a \quad (w \in \mathbb{R}) \\
a^{\frac{p}{q}} &= \sqrt[q]{a^p} \\
a^{-\frac{p}{q}} &= \tfrac{1}{a^{\frac{p}{q}}}.
\end{aligned}$$

Voorbeelden

- $2^3 = 2 \cdot 2 \cdot 2 = 8$
- $2^1 = 2$

- $2^0 = 1$
- $2^{-3} = \frac{1}{2^3} = \frac{1}{8}$
- $2^{-1} = \frac{1}{2^1} = \frac{1}{2}$
- $2^{\frac{3}{4}} = \sqrt[4]{2^3} = \sqrt[4]{8}$ ($\approx 1{,}6818$)
- $2^{-\frac{3}{4}} = \frac{1}{2^{\frac{3}{4}}} = \frac{1}{\sqrt[4]{2^3}} = \frac{1}{\sqrt[4]{8}}$ ($\approx 0{,}5946$)

Uit de definitie van de machtsverheffing kunnen de volgende rekenregels afgeleid worden:

Rekenregels
Stel $a, b \in \mathbb{R}_0^+$. Voor alle $r, s \in \mathbb{Q}$ geldt

$$\begin{aligned} a^r \cdot a^s &= a^{r+s} \\ \frac{a^r}{a^s} &= a^{r-s} \\ (a^r)^s &= a^{r \cdot s} \\ a^r \cdot b^r &= (a \cdot b)^r \\ \frac{a^r}{b^r} &= (\frac{a}{b})^r. \end{aligned}$$

Voorbeelden

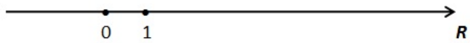
- $2^3 \cdot 2^5 = 2^{3+5} = 2^8 = 256$
- $\frac{2^3}{2^5} = 2^{3-5} = 2^{-2} = \frac{1}{2^2} = \frac{1}{4}$
- $(2^3)^5 = 2^{3 \cdot 5} = 2^{15} = 32\,768$
- $2^3 \cdot 5^3 = (2 \cdot 5)^3 = 10^3 = 1000$
- $\frac{10^3}{5^3} = (\frac{10}{5})^3 = 2^3 = 8$

7.3 Intervallen in $\mathbb{R}$

De verzameling $\mathbb{R}$ kan voorgesteld worden als een rechte, zie Figuur 7.1. Eens op deze reële rechte een ijk is aangebracht kan om het even welk reëel getal hierop aangeduid worden.

Elk punt van de reële rechte komt overeen met één specifiek reëel getal.

Wanneer we een aaneengesloten deel van de reële rechte willen aanduiden, spreken we van een *interval*. Een interval is een deelverzameling van $\mathbb{R}$. We maken onderscheid tussen vier soorten intervallen:



Figuur 7.1: De reële rechte $\mathbb{R}$.

- Open interval:
  bevat alle reële getallen gelegen tussen twee reële waarden a en b, a en b zelf *niet* inbegrepen (Figuur 7.2).

$$]a, b[= \{x \in \mathbb{R} \,|\, a < x < b\} \subseteq \mathbb{R}$$

Figuur 7.2: Het open interval $]a, b[$.

- Gesloten interval:
  bevat alle reële getallen gelegen tussen twee reële waarden a en b, a en b inbegrepen (Figuur 7.3) .

$$[a, b] = \{x \in \mathbb{R} \,|\, a \leq x \leq b\} \subseteq \mathbb{R}$$

Figuur 7.3: Het gesloten interval $[a, b]$.

- Halfopen interval:
  Het halfopen interval $[a, b[$ bevat alle reële getallen gelegen tussen twee reële waarden a en b, a inbegrepen, b niet (Figuur 7.4).

$$[a, b[= \{x \in \mathbb{R} \,|\, a \leq x < b\} \subseteq \mathbb{R}$$

  Het halfopen interval $]a, b]$ bevat alle reële getallen gelegen tussen twee reële waarden a en b, a niet inbegrepen, b wel inbegrepen (Figuur 7.5).

$$]a, b] = \{x \in \mathbb{R} \,|\, a < x \leq b\} \subseteq \mathbb{R}$$

Figuur 7.4: Het halfopen interval $[a, b[$.

Figuur 7.5: Het halfopen interval $]a, b]$.

- De verzameling van de reële getallen kan zelf ook als een interval voorgesteld worden: $\mathbb{R}$ bevat alle reële getallen gelegen tussen $-\infty$ en $+\infty$, $-\infty$ en $+\infty$ zelf niet inbegrepen, dus
$$\mathbb{R} =]-\infty, +\infty[.$$
De reële rechte is reeds voorgesteld in Figuur 7.1.

Deze intervalnotatie voor $\mathbb{R}$ geeft duidelijk weer dat oneindig (∞) niet tot de reële getallen behoort. Nochtans wordt binnen de analyse vaak gewerkt met het begrip oneindig. Wanneer de reële rechte uitgebreid wordt met $-\infty$ en $+\infty$, dan spreken we van de *uitgebreide reële rechte*.

7.4 Het begrip oneindig in de wiskunde

7.4.1 De uitgebreide reële rechte

De verzameling $\mathbb{R}$ is de verzameling van alle reële getallen. Zoals reeds aangegeven maken $-\infty$ en $+\infty$ geen deel uit van deze verzameling. Door beide elementen toe te voegen aan $\mathbb{R}$ bekomen we de uitgebreide reële rechte $\overline{\mathbb{R}}$.

Definitie 7.2 *De **uitgebreide reële rechte** $\overline{\mathbb{R}}$ is:*

$$\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, +\infty\} = [-\infty, +\infty]$$

met $-\infty < x < +\infty$ voor elke $x \in \mathbb{R}$.

7.4.2 Eigenschappen

Wanneer we de verzameling $\mathbb{R}$ uitbreiden met de waarden $-\infty$ en $+\infty$ tot $\overline{\mathbb{R}}$, is het eveneens nodig rekenregels toe te voegen die toelaten om te rekenen met $-\infty$ en $+\infty$.

Rekenregels voor optellen en aftrekken
Voor elke $x \in \mathbb{R}$ geldt

$$\begin{aligned} +\infty + x &= +\infty \\ -\infty + x &= -\infty \\ +\infty + (+\infty) &= +\infty \\ -\infty + (-\infty) &= -\infty. \end{aligned}$$

Let op! De volgende bewerkingen zijn niet gedefinieerd. Er is geen zinvolle uitkomst voor

$$\begin{array}{l} +\infty + (-\infty) \\ -\infty + (+\infty). \end{array}$$

Rekenregels voor vermenigvuldigen en delen
Voor elke $x \in \mathbb{R}_0^+$ geldt

$$\begin{aligned} +\infty \cdot x &= +\infty \\ -\infty \cdot x &= -\infty \\ +\infty / x &= +\infty \\ -\infty / x &= -\infty. \end{aligned}$$

Voor elke $x \in \mathbb{R}_0^-$ geldt

$$\begin{aligned} +\infty \cdot x &= -\infty \\ -\infty \cdot x &= +\infty \\ +\infty / x &= -\infty \\ -\infty / x &= +\infty. \end{aligned}$$

Voor elke $x \in \mathbb{R}$ geldt

$$\begin{aligned} x/(+\infty) &= 0 \\ x/(-\infty) &= 0. \end{aligned}$$

Voor elke $x \in \mathbb{R}_0$ geldt

$$x/0 = \infty.$$

Dit laatste betekent dat $x/0$ ofwel $+\infty$ ofwel $-\infty$ is al naargelang de situatie.

Eveneens geldt

$$\begin{aligned} +\infty \cdot (+\infty) &= +\infty \\ -\infty \cdot (-\infty) &= +\infty \\ +\infty \cdot (-\infty) &= -\infty \\ -\infty \cdot (+\infty) &= -\infty. \end{aligned}$$

Let op! De volgende bewerkingen zijn niet gedefinieerd:

$$\begin{array}{l} 0 \cdot (+\infty) \\ 0 \cdot (-\infty) \\ \infty / \infty \\ 0/0 \\ 0^0 \\ \infty^0. \end{array}$$

Voorbeelden

- $-1000 + (+\infty) = +\infty$
- $-\infty \cdot 1000 = -\infty$
- $\frac{-\infty}{-1000} = +\infty$
- $\frac{-1000}{+\infty} = 0$

7.5 Extra oefeningen

1. Stel $a \in \mathbb{R}_0$. Schrijf de volgende uitdrukkingen zo eenvoudig mogelijk.
   a) $a^3 + a^3$
   b) $a^3 \cdot a^3$
   c) $(a^3)^3$
   d) $a^3 \cdot a^{-3}$
   e) $(3 \cdot a)^3$
   f) $\dfrac{a^3}{a^{-3}}$
   g) $(\dfrac{3}{a})^{-3}$
2. Vereenvoudig. $(x, y \in \mathbb{R}_0)$
   a) $\dfrac{(-2x^4y)^3}{(-2x^3y)^2}$
   b) $(-x^2y^3)^4 \cdot (-2xy^4)^{-3}$
   c) $x(x-1) + (3x+1)(x-2)$

7.6 Oplossingen

1. a) $a^3 + a^3 = 2 \cdot a^3$
   b) $a^3 \cdot a^3 = a^6$
   c) $(a^3)^3 = a^9$
   d) $a^3 \cdot a^{-3} = 1$
   e) $(3 \cdot a)^3 = 27 \cdot a^3$
   f) $\dfrac{a^3}{a^{-3}} = a^6$

g) $(\frac{3}{a})^{-3} = \frac{a^3}{27}$

2. a) $\frac{(-2x^4y)^3}{(-2x^3y)^2} = -2x^6y$

   b) $(-x^2y^3)^4 \cdot (-2xy^4)^{-3} = -\frac{x^5}{8}$

   c) $x(x-1) + (3x+1)(x-2) = 4x^2 - 6x - 2$

HOOFDSTUK **8**

Veeltermfuncties

In deel 1: 'Verzamelingen, Relaties en Functies' wordt het begrip functie algemeen besproken.
In dit deel werken we met *reële functies*. Dit zijn functies waarvoor zowel het *argument* als het *beeld* gekozen worden uit de verzameling van de reële getallen.

Verder beperken we ons tot reële functies in één variabele. Deze variabele wordt standaard aangeduid met het symbool x.

Definitie 8.1 *Een functie f is een* ***reële functie*** *indien zijn bron- en doelverzameling beide $\mathbb{R}$ zijn. D.w.z. dat voor elke $x \in \mathbb{R}$ hoogstens één $y \in \mathbb{R}$ bestaat zodanig dat $f(x) = y$.*
We noemen x het ***argument*** *terwijl y het* ***beeld*** *wordt genoemd.*

Notaties
$f : \mathbb{R} \to \mathbb{R} : x \mapsto y$
$\mathcal{F}(\mathbb{R}, \mathbb{R})$ = de verzameling van alle reële functies.

Reële functies worden opgedeeld in verschillende subgroepen. De *veeltermfuncties* vormen zo'n subgroep.

Als bijzondere veeltermfuncties zullen we de constante functie, de eerste- en tweedegraadsfuncties bespreken.

8.1 Definities en notaties

Alvorens de veeltermfuncties in detail te benaderen eerst nog een aantal belangrijke definities en notaties i.v.m. reële functies.

Reële functies worden ook opgedeeld volgens het uitzicht van hun grafiek. Een functie is *continu* wanneer zijn grafiek uit één 'stuk' bestaat.

Definitie 8.2 *Een intuïtieve definitie:*

- *Een reële functie f is **continu**, indien we de grafiek van f kunnen tekenen zonder ons potlood op te heffen.*
- *Een reële functie f is **continu in een punt** a van haar domein, indien de grafiek van f geen 'sprong' vertoont in de onmiddellijke omgeving van het punt $(a, f(a))$. In het ander geval spreekt men van een **discontinuïteit in het punt** a.*

De x-waarden waarvoor geldt $f(x) = 0$ worden de *nulpunten* van de functie f genoemd. Een functie kan meerdere nulpunten hebben.

Definitie 8.3 *Een **nulpunt** x van een reële functie f is een element van het domein van f waarvoor de functiewaarde gelijk is aan 0. M.a.w. een nulpunt x van een functie f is oplossing van de vergelijking $f(x) = 0$.*

Een nulpunt van een functie f komt dus steeds overeen met de x-coördinaat van een snijpunt van de grafiek van f met de X-as (de rechte met vergelijking $y = 0$).
Het snijpunt met de Y-as (de rechte met vergelijking $x = 0$) is steeds het punt $(0, f(0))$. Dit betekent dat er meerdere snijpunten met de X-as kunnen zijn maar hoogstens één snijpunt met de Y-as.

Voorbeeld Stel

$$f : \mathbb{R} \to \mathbb{R} : x \mapsto y = 5x - 3.$$

Bepaal de snijpunten met X en Y-as voor de functie f.

- Het nulpunt is: $x = \frac{3}{5}$.
  Verklaring: $f(x) = 0 \Leftrightarrow 5x - 3 = 0 \Leftrightarrow x = \frac{3}{5}$.
  Het snijpunt met de X-as is dus $(\frac{3}{5}, 0)$.
- Het snijpunt met de Y-as is: $(0, -3)$.
  Verklaring: $f(0) = 5 \cdot 0 - 3 = -3$.

Definitie 8.4 *Een **veeltermfunctie** is een reële functie met als functievoorschrift*

$$f : \mathbb{R} \to \mathbb{R} : x \mapsto f(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0,$$

met $n \in \mathbb{N}$ en $a_i \in \mathbb{R}$ ($i = 0 \ldots n$), $a_n \neq 0$.
*We noemen n de **graad** van $f(x)$ en $a_n x^n$ de **hoogste graadsterm**.*

Voorbeelden

- Beschouw de functie f

  $$f : \mathbb{R} \rightarrow \mathbb{R} : x \mapsto y = 5x - 3.$$

  De functie f is een veeltermfunctie van de eerste graad.

- Beschouw de functie g

  $$g : \mathbb{R} \rightarrow \mathbb{R} : x \mapsto g(x) = x^3 - x - 2.$$

  De functie g is een veeltermfunctie van de derde graad met x^3 als hoogste graadsterm.

Het domein van om het even welke veeltermfunctie is steeds de volledige verzameling $\mathbb{R}$. Er geldt immers $f(x) \in \mathbb{R}$ voor elke $x \in \mathbb{R}$.

We bestuderen een aantal specifieke veeltermfuncties.

8.2 Constante functies

8.2.1 Algemeen

Een constante functie f is een veeltermfunctie van de 0-de graad. Dit betekent dat de hoogste graadsterm van de veelterm $f(x)$ van de vorm a_0x^0 is met $a_0 \in \mathbb{R}$.

Definitie 8.5 *Een **constante functie** is een functie waarbij de functiewaarde constant is.*

$$f : \mathbb{R} \rightarrow \mathbb{R} : x \mapsto f(x) = a \qquad (a \in \mathbb{R}).$$

Grafiek

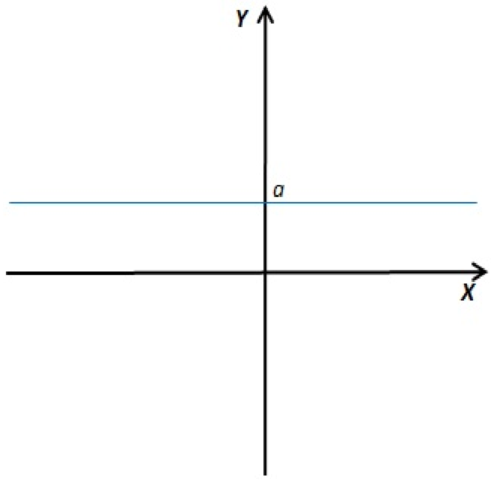
De grafiek van een dergelijke functie $f : \mathbb{R} \rightarrow \mathbb{R} : x \mapsto a$ $(a \in \mathbb{R})$ is steeds een rechte met vergelijking $y = a$. Dit is een rechte evenwijdig met de X-as (zie Figuur 8.1).

Eigenschappen

- Het domein:
  $\text{dom}(f) = \mathbb{R}$.

- Het beeld:
  $\text{bld}(f) = \{a\}$.

- De constante functie f is continu in $\mathbb{R}$.



Figuur 8.1: De grafiek van de constante functie $y = a$.

- Nulpunten:
  - als $a \neq 0$ dan zijn er geen nulpunten;
  - als $a = 0$ dan zijn er oneindig veel nulpunten: voor alle $x \in \mathbb{R}$ geldt $f(x) = 0$.
- Snijpunt met de Y-as:
  $(0, a)$
  (verklaring: $f(0) = a$).

Tekenverloop

De constante functie $f : x \mapsto a$ is

- positief in gans $\mathbb{R}$ als en slechts als $a > 0$.
- nul in gans $\mathbb{R}$ als en slechts als $a = 0$.
- negatief in gans $\mathbb{R}$ als en slechts als $a < 0$.

Samengevat in een **tekentabel** wordt dit:

x	$-\infty$ $\qquad\qquad\qquad$ $+\infty$
$f(x)$	teken van a

8.2.2 Voorbeeld

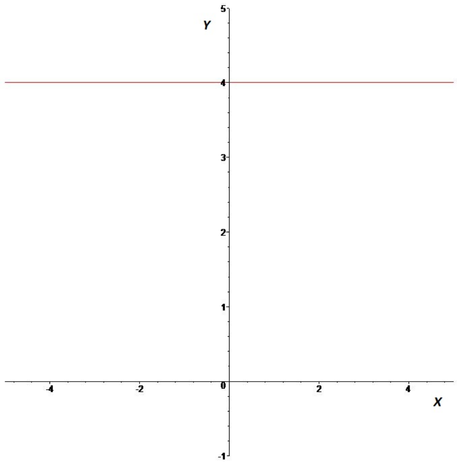
Beschouw de functie f

$$f : \mathbb{R} \to \mathbb{R} : x \mapsto 4.$$

- **Eigenschappen**
  - $\text{dom}(f) = \mathbb{R}$
  - $\text{bld}(f) = \{4\}$
  - f heeft geen nulpunten
  - snijpunt met de Y-as: $(0, 4)$
- **Tekenonderzoek**

x	$-\infty$	$+\infty$
$f(x)$	$+$	

- **Grafiek**
  De grafiek wordt weergegeven in Figuur 8.2.



Figuur 8.2: Grafiek van de rechte met vergelijking $y = 4$.

8.2.3 Oefening

De constante functie f bevat het koppel $(2, -3)$. Bepaal deze functie f.

8.3 Lineaire functies of functies van de eerste graad

8.3.1 Algemeen

Een lineaire functie is een veeltermfunctie van de eerste graad.

Definitie 8.6 *De functie $f : \mathbb{R} \to \mathbb{R} : x \mapsto ax + b$ waarbij a en b gegeven reële getallen zijn en $a \neq 0$, noemt men een **functie van de eerste graad** of **lineaire functie**.*

Grafiek

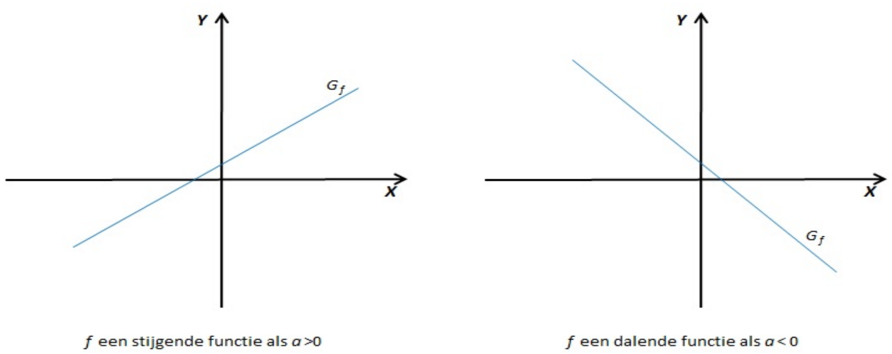
De grafiek van de eerstegraadsfunctie $f : x \mapsto ax + b$ is de rechte met vergelijking $y = ax + b$.
Het is een rechte die de X-as snijdt. Het is een schuine rechte, d.w.z. niet evenwijdig met de Y-as.
De x-coördinaat van het snijpunt met de X-as noemen we het **nulpunt** van de functie en is de oplossing van de vergelijking $ax + b = 0$.
De coëfficiënt a bepaalt de mate waarin de rechte stijgt of daalt en wordt de **richtingscoëfficiënt (rico)** van de rechte genoemd.

Eigenschappen

- Het domein:
  $\text{dom}(f) = \mathbb{R}$.
- Het beeld:
  $\text{bld}(f) = \mathbb{R}$.
- De lineaire functie f is continu in $\mathbb{R}$.
- Nulpunt:
  $x \in \mathbb{R}$ waarvoor geldt $f(x) = 0$, i.e. $x = -\dfrac{b}{a}$.

Tekenverloop

Een eerstegraadsfunctie $f : x \mapsto ax + b$ is stijgend in $\mathbb{R}$ als $a > 0$, met a de rico van f.
Een eerstegraadsfunctie $f : x \mapsto ax + b$ is dalend in $\mathbb{R}$ als $a < 0$, met a de rico van f.
Dit wordt geïllustreerd in Figuur 8.3.



Figuur 8.3: Grafiek van een stijgende en een dalende rechte.

De bijhorende tekentabel is

x	$-\infty$		$-\frac{b}{a}$		$+\infty$
$f(x)$		teken van $-a$	0	teken van a	

8.3.2 Voorbeeld

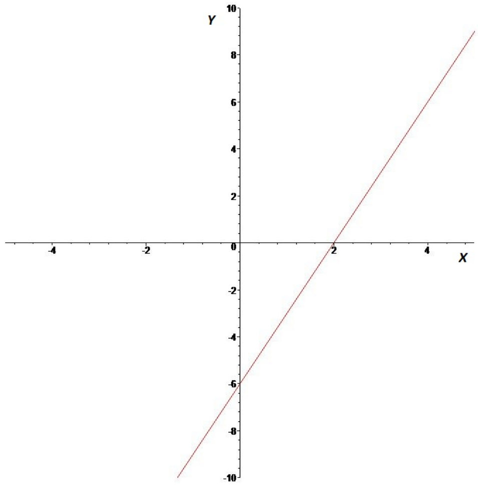
Beschouw de functie

$$f : \mathbb{R} \to \mathbb{R} : x \mapsto 3x - 6.$$

- **Eigenschappen**
  - $\text{dom}(f) = \mathbb{R}$.
  - $\text{bld}(f) = \mathbb{R}$.
  - nulpunt: $x = 2$.
    Verklaring: $3x - 6 = 0 \Leftrightarrow 3x = 6 \Leftrightarrow x = 2$.
  - snijpunt met Y-as: $(0, -6)$.
    Verklaring: $f(0) = 3 \cdot 0 - 6 = -6$.
  - rico $= 3 > 0$,
    de grafiek van f is een stijgende rechte.
- **Tekenonderzoek**
  De tekentabel kan pas correct opgesteld worden wanneer het nulpunt van de functie gekend is. Dit nulpunt hebben we zonet berekend (zie eigenschappen). Als resultaat hebben we gevonden: $f(x) = 0 \Leftrightarrow x = 2$.
  Combineren we dit resultaat met het feit dat de grafiek van f een stijgende rechte is, dan vinden we:

x	$-\infty$		2		$+\infty$
$f(x)$		$-$	0	$+$	

- **Grafiek**
  Twee punten bepalen een rechte. Verbinden we het snijpunt met de X-as, $(2, 0)$, en het snijpunt met de Y-as, $(0, -6)$, door een rechte lijn dan hebben we de grafiek van f getekend.
  De grafiek wordt voorgesteld in Figuur 8.4. Het resultaat is inderdaad een stijgende rechte.



Figuur 8.4: De grafiek van de functie $f : \mathbb{R} \to \mathbb{R} : x \mapsto 3x - 6$.

8.3.3 De vergelijking van een rechte door twee gegeven punten

Er is slechts één rechte die door twee verschillende gegeven punten gaat.
Wanneer twee verschillende punten van een rechte gekend zijn, dan is het perfect mogelijk om a.d.h.v. deze gegeven punten het functievoorschrift van de rechte op te stellen.

Stel $(x_1, y_1) \in f$ en $(x_2, y_2) \in f$, met f een lineaire functie, dan geldt:

$$f : \mathbb{R} \to \mathbb{R} : x \mapsto y = \frac{y_2 - y_1}{x_2 - x_1} \cdot (x - x_1) + y_1$$

met $\frac{y_2-y_1}{x_2-x_1}$ de **richtingscoëfficiënt (rico)** van de rechte.

Voorbeeld
Zij f een eerste graadsfunctie die door de punten (3,4) en (5,8) gaat. Er geldt

$$\begin{aligned} y = \tfrac{y_2-y_1}{x_2-x_1} \cdot (x - x_1) + y_1 \quad &\Leftrightarrow \quad y = \tfrac{8-4}{5-3} \cdot (x-3) + 4 \\ &\Leftrightarrow \quad y = 2 \cdot (x-3) + 4 \\ &\Leftrightarrow \quad y = 2x - 6 + 4 \\ &\Leftrightarrow \quad y = 2x - 2. \end{aligned}$$

Of nog

$$f : \mathbb{R} \to \mathbb{R} : x \mapsto y = 2x - 2.$$

8.3.4 De vergelijking van een rechte door één gegeven punt en gegeven rico

Wanneer de richtingscoëfficiënt van een rechte f (rico) en één bepaald punt (x_1, y_1), gelegen op deze rechte, gekend zijn, kan de vergelijking van de rechte f als volgt bepaald worden:

$$f : \mathbb{R} \to \mathbb{R} : x \mapsto y = \text{rico} \cdot (x - x_1) + y_1.$$

Voorbeeld
Zij g een rechte met rico = -2 en $(3, 1) \in g$.
Gelet op de voorgaande formule vinden we:

$$\begin{aligned} y = \text{rico} \cdot (x - x_1) + y_1 \quad &\Leftrightarrow \quad y = -2 \cdot (x - 3) + 1 \\ &\Leftrightarrow \quad y = -2x + 6 + 1 \\ &\Leftrightarrow \quad y = -2x + 7 \end{aligned}$$

en dus

$$g : \mathbb{R} \to \mathbb{R} : x \mapsto y = -2x + 7.$$

8.3.5 Oefeningen

1. Onderzoek het verloop van de volgende functies van de eerste graad. Teken de grafiek van de functie.

   a) $f : x \mapsto x - 3.$

   b) $g : x \mapsto -\frac{3}{5}x + 7.$

2. Bepaal het snijpunt van de grafieken van

   a) $f : x \mapsto -3x - 2$ en $g : x \mapsto 5x + 6.$

   b) $f : x \mapsto -\frac{2}{3}x + \frac{1}{2}$ en $g : x \mapsto \frac{7}{6}x - \frac{1}{6}.$

   c) $f : x \mapsto x\sqrt{3} + 6\sqrt{3}$ en $g : x \mapsto -2x\sqrt{3}.$

3. (Bron [De Pauw et al., 2009]) Een object beweegt tijdens een computerspel langs een rechte lijn van het punt A met coördinaten $(0, 20)$ naar het punt B met coördinaten $(15, 30)$. Zoek de vergelijking van deze rechte. Als het voorwerp in het punt C met coördinaten $(30, 40)$ komt, beweegt de speler de joystick, zodat het object 90° naar links draait en in een rechte lijn verder beweegt. Vind de vergelijking van de rechte die het nieuwe pad voorstelt. Teken bovendien beide rechten in één assenstelsel.
   Tip: Twee rechten f en g, de rechte f met rico $= r_1$ en de rechte g met rico $= r_2$, staan loodrecht op elkaar enkel en alleen als $r_1 \cdot r_2 = -1$.

8.4 Functies van de tweede graad

8.4.1 Algemeen

Tweedegraadsfuncties zijn deze waarvoor het functievoorschrift een veelterm van de tweede graad voorstelt. De grafiek die correspondeert met een dergelijke functie is een parabool.

Definitie 8.7 *Een functie $f : \mathbb{R} \to \mathbb{R} : x \mapsto ax^2 + bx + c$ waarbij a, b en c gegeven reële getallen zijn en waarvoor $a \neq 0$, noemt men een functie van de tweede graad.*

Grafiek

De grafiek van de functie $f : x \mapsto ax^2 + bx + c$ is een **parabool** P.
De verticale rechte met vergelijking $x = -\frac{b}{2a}$ is de **symmetrie-as**. Het snijpunt van de symmetrie-as en de parabool is de **top**. Dit is het punt met coördinaten $(-\frac{b}{2a}, f(-\frac{b}{2a}))$.
Als $a > 0$ is P een dalparabool.
Als $a < 0$ is P een bergparabool.

Domein

Voor een tweedegraadsfunctie geldt steeds

$$\text{dom}(f) = \mathbb{R}.$$

Beeld

Voor de omschrijving van het beeld moeten we onderscheid maken tussen een dalparabool en een bergparabool.

- Dalparabool:
  $\text{bld}(f) = [f(-\frac{b}{2a}), +\infty[$.
- Bergparabool:
  $\text{bld}(f) =]-\infty, f(-\frac{b}{2a})]$.

Nulpunten

De **nulpunten** van $f : x \mapsto ax^2 + bx + c$ zijn de waarden van x waarvoor $f(x) = 0$, m.a.w. de eerste coördinaatgetallen van de snijpunten van de grafiek van f met de X-as.

Voor de berekening van de nulpunten moeten we de vergelijking $ax^2 + bx + c = 0$ oplossen naar x. Hiervoor bestaan verschillende oplossingsmethodes. Een methode die altijd werkt bij elke tweedegraadsvergelijking is de oplossingsmethode met behulp van de discriminant D.
We herhalen even de formules:

$$D = b^2 - 4ac.$$

De waarde van de discriminant D geeft aan of er $0, 1$ of 2 nulpunten zijn:

- $D > 0$: er zijn twee verschillende nulpunten
$$x_{1,2} = \frac{-b \pm \sqrt{D}}{2a}.$$
- $D = 0$: er is juist één nulpunt
$$x_1 = x_2 = -\frac{b}{2a}.$$
- $D < 0$: er is geen nulpunt: de parabool ligt volledig boven of onder de X-as.

Eens de wortels (= nulpunten) gekend zijn, kunnen we de kwadratische vergelijking als volgt ontbinden in factoren

$$ax^2 + bx + c = a(x - x_1)(x - x_2).$$

Asymptoten

Een tweedegraadsfunctie heeft geen asymptoten.

Extrema

Een tweedegraadsfunctie heeft één extremum, nl. de top van de parabool. Bij een dalparabool is dit extremum een minimum, bij een bergparabool een maximum.

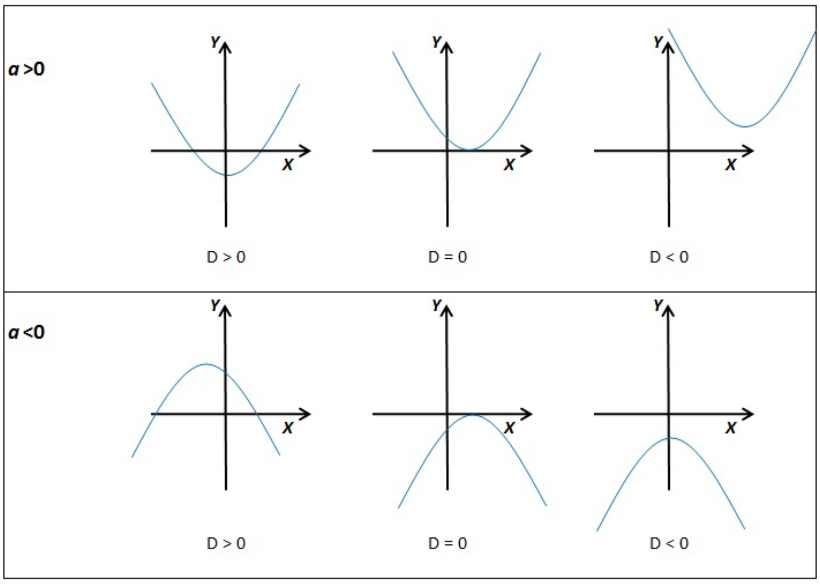
Tekenonderzoek

We hebben reeds vermeld dat voor $a > 0$ de grafiek een dalparabool is, terwijl voor $a < 0$ de grafiek een bergparabool is. Als we deze informatie koppelen aan de waarde van de discriminant D die ons vertelt hoeveel nulpunten er zijn, dan kunnen we in totaal zes verschillende situaties onderscheiden voor de grafiek van een tweedegraadsfunctie. Deze worden allen voorgesteld in Figuur 8.5.

Voor het tekenonderzoek maken we onderscheid tussen de drie verschillende situaties: $D > 0$, $D = 0$ en $D < 0$. De nulpunten zijn x_1 en x_2 (indien aanwezig).

- $D > 0$

x	$-\infty$		x_1		x_2		$+\infty$
$f(x)$		teken van a	0	teken van $-a$	0	teken van a	



Figuur 8.5: Illustratie van het verband tussen de grafiek van een tweedegraadsfunctie en het teken van a en D.

- $D = 0$

x	$-\infty$		x_1		$+\infty$
$f(x)$		teken van a	0	teken van a	

- $D < 0$

x	$-\infty$		$+\infty$
$f(x)$		teken van a	

8.4.2 Voorbeeld

Beschouw de functie

$$f : \mathbb{R} \to \mathbb{R} : x \mapsto 3x^2 - 4x + 1.$$

- **Domein en beeld**
  - $\text{dom}(f) = \mathbb{R}$.
  - $\text{bld}(f) = [-\frac{1}{3}, +\infty[\quad (\text{want } f(\text{top}) = -\frac{1}{3})$.

- **Nulpunten**
Voor het bepalen van de nulpunten kan gewerkt worden met de discriminant:

$$D = b^2 - 4ac = (-4)^2 - 4 \cdot 3 \cdot 1 = 16 - 12 = 4 > 0.$$

Er zijn twee nulpunten x_1 en x_2:

$$x_{1,2} = \frac{-b \pm \sqrt{D}}{2a} = \frac{4 \pm \sqrt{4}}{6} = \frac{4 \pm 2}{6}$$

en dus

$$x_1 = \frac{4-2}{6} = \frac{1}{3}$$

en

$$x_2 = \frac{4+2}{6} = 1.$$

- **Snijpunt met Y-as**
$(0, 1)$ (verklaring: $f(0) = 1$).

- **Top (cfr. extremum)**
Voor de top $t(t_x, t_y)$ geldt

$$t_x = \frac{-b}{2a} = \frac{4}{6} = \frac{2}{3}$$

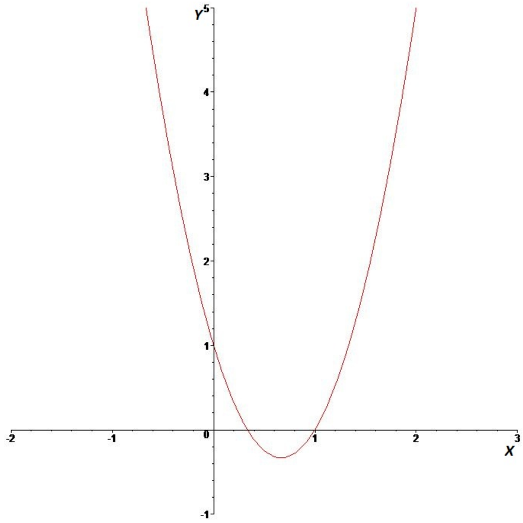
en

$$t_y = f(t_x) = 3(\frac{2}{3})^2 - 4 \cdot \frac{2}{3} + 1 = 3 \cdot \frac{4}{9} - \frac{8}{3} + 1 = \frac{4}{3} - \frac{8}{3} + 1 = -\frac{1}{3}.$$

- **Tekenonderzoek**
Eens de nulpunten gekend zijn is het mogelijk de tekentabel voor f op te stellen:

x	$-\infty$		$\frac{1}{3}$		1		$+\infty$
$f(x)$		$+$	0	$-$	0	$+$	

- **Grafiek**
De tekentabel samen met de coördinaten van de top laten ons toe een schets te maken van de grafiek van f. Deze dalparabool ($a > 0$) wordt weergegeven in Figuur 8.6.



Figuur 8.6: De grafiek van $f : \mathbb{R} \rightarrow \mathbb{R} : x \mapsto 3x^2 - 4x + 1$.

8.4.3 Oefeningen

1. Bespreek de volgende tweedegraadsfuncties:

   a) $f : x \mapsto -x^2 + 2x + 5$.

   b) $f : x \mapsto 3x^2 + 2x + 5$.

   c) $f : x \mapsto -x^2 + 2x - 1$.

   d) $f : x \mapsto 3x^2 + 4x$.

   e) $f : x \mapsto x^2 - 9$.

2. Is de inverse van een tweedegraadsfunctie opnieuw een functie?
   Illustreer dit voor de functie $f : x \mapsto f(x) = x^2$.

8.5 Extra oefeningen

1. Gegeven

$$f : \mathbb{R} \rightarrow \mathbb{R} : x \mapsto f(x) = -2x^2 - 2x + 4$$
$$g : \mathbb{R} \rightarrow \mathbb{R} : x \mapsto g(x) = -4x$$

   Gevraagd:

   a) Bepaal het domein en beeld van f. Geef de nulpunten en het snijpunt met de Y-as.
   Idem voor de functie g.

b) Teken de grafiek van f en g. Duid de snijpunten aan op de grafiek.

c) Bereken de snijpunten van f en g analytisch.

d) Geef het functievoorschrift van de volgende functies

   i. $f \circ g$

   ii. $g \circ f$

   iii. g^{-1}

8.6 Oplossingen

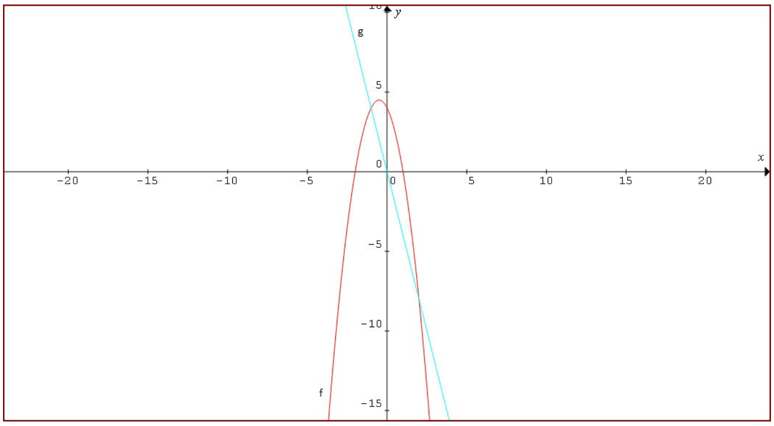
1. a) Voor f vinden we:

   - $\operatorname{dom} f = \mathbb{R}$.
   - $\operatorname{bld} f =]-\infty, \frac{9}{2}]$.
   - nulpunten: $x \in \{-2, 1\}$.
   - snijpunt met de Y-as: $(0, 4)$.

   Voor g vinden we:

   - $\operatorname{dom} g = \mathbb{R}$.
   - $\operatorname{bld} g = \mathbb{R}$.
   - nulpunt: $x = 0$.
   - snijpunt met de Y-as: $(0, 0)$.

   b) Grafiek:



Figuur 8.7: De grafiek van de functie f en g.

   c) De functies f en g snijden in de punten $(-1, 4)$ en $(2, -8)$.

   d) De gevraagde functievoorschriften:

   i. $f \circ g : \mathbb{R} \to \mathbb{R} : x \mapsto f(g(x)) = -32x^2 + 8x + 4$

ii. $g \circ f : \mathbb{R} \to \mathbb{R} : x \mapsto g(f(x)) = 8x^2 + 8x - 16$

iii. $g^{-1} : \mathbb{R} \to \mathbb{R} : x \mapsto g^{-1}(x) = -\frac{1}{4}x$

HOOFDSTUK 9

De exponentiële en logaritmische functie

In dit hoofdstuk bespreken we twee bijzondere functies: de exponentiële functie en haar inverse de logaritmische functie.

9.1 De exponentiële functie

Een exponentiële functie is een functie waarbij een *vast grondtal* tot een *variabele macht* wordt verheven. Het functievoorschrift van een dergelijke functie komt overeen met een machtsverheffing.

Definitie 9.1 *De* ***exponentiële functie met grondtal*** a $(a \in \mathbb{R}_0^+ \setminus \{1\})$ *is*

$$f : \mathbb{R} \to \mathbb{R} : x \mapsto a^x.$$

De restrictie voor het grondtal a, nl. $a \in \mathbb{R}_0^+ \setminus 1$, garandeert dat elke exponentiële functie strikt stijgend of strikt dalend is.

Voor $a > 1$ is de exponentiële functie strikt stijgend.
Voor $0 < a < 1$ is de exponentiële functie strikt dalend.

9.1.1 Voorbeelden

We bekijken twee voorbeelden.
Het eerste voorbeeld is een illustratie van een stijgende exponentiële functie. Het tweede voorbeeld betreft een exponentiële functie met grondtal kleiner dan 1.

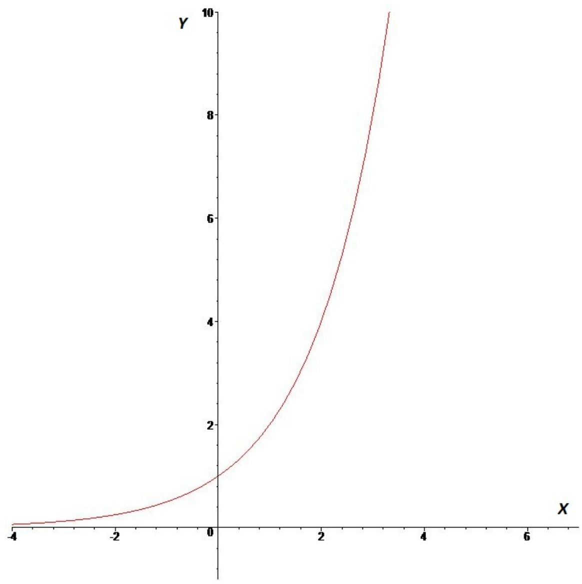
Voorbeeld 1: de exponentiële functie f met grondtal $a = 2$.

$$f : \mathbb{R} \to \mathbb{R} : x \mapsto f(x) = 2^x$$

De volgende punten behoren tot de functie f.

x	$f(x)$
-2	$2^{-2} = \frac{1}{2^2} = \frac{1}{4}$
-1	$2^{-1} = \frac{1}{2}$
0	$2^0 = 1$
1	$2^1 = 2$
2	$2^2 = 4.$

Zetten we deze punten uit in een cartesiaans assenstelsel en verbinden we ze door een vloeiende kromme dan krijgen we de grafiek zoals weergegeven in Figuur 9.1.



Figuur 9.1: Grafiek van de exponentiële functie met grondtal 2.

De grafiek van f is een strikt stijgende continue functie in $\mathbb{R}$. M.a.w. als de inputwaarde x groter wordt, wordt de outputwaarde y ook groter.

Het domein van f is $\mathbb{R}$.

Aangezien $\infty \notin \mathbb{R}$, is $f(\infty)$ niet gedefinieerd.
Willen we toch uitdrukken wat er gebeurt in de omgeving van oneindig dan kan dit a.d.h.v. een **limiet**. We zeggen: de limiet van de functie f voor x naderend naar $+\infty$ is $+\infty$. Dit betekent, hoe groter x wordt aan de positieve kant hoe groter de functiewaarde y zal worden. In symbolen wordt dit als volgt genoteerd: $\lim\limits_{x \to +\infty} f(x) = +\infty$.

Op analoge wijze kunnen we bespreken wat er gebeurt aan de kant van $-\infty$. Op de grafiek zien we: hoe kleiner x wordt, aan de negatieve kant, hoe dichter y nadert naar 0. Of nog, $\lim\limits_{x \to -\infty} f(x) = 0$. Aan de negatieve kant nadert de grafiek van f steeds dichter naar de X-as. Een dergelijke rechte, waar de grafiek van een functie naar nadert, noemen we een **asymptoot**.

In dit specifiek geval nadert de grafiek naar één specifieke rechte, nl. de X-as, als x nadert naar $-\infty$. We noemen de X-as een **horizontale asymptoot** van de functie f omdat de X-as een horizontale rechte is.

Voor elke $x \in \mathbb{R}$ geldt steeds $f(x) \in \mathbb{R}_0^+$, dus $\text{bld} f = \mathbb{R}_0^+$.

Aangezien de grafiek van de functie f de X-as nooit bereikt, heeft f geen nulpunten. Het snijpunt met de Y-as is $(0, 1)$, immers $f(0) = 2^0 = 1$.

Voorbeeld 2: de exponentiële functie f met grondtal $a = \frac{1}{2}$.

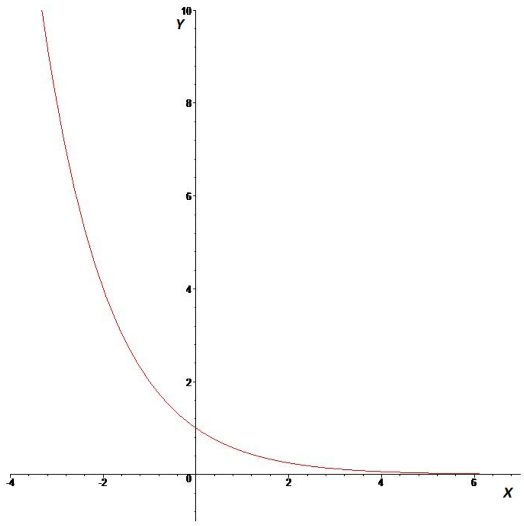
$$f : \mathbb{R} \to \mathbb{R} : x \mapsto f(x) = (\frac{1}{2})^x$$

De volgende punten behoren tot de functie f.

x	$f(x)$
-2	$(\frac{1}{2})^{-2} = 2^2 = 4$
-1	$(\frac{1}{2})^{-1} = 2^1 = 2$
0	$(\frac{1}{2})^0 = 1$
1	$(\frac{1}{2})^1 = \frac{1}{2}$
2	$(\frac{1}{2})^2 = \frac{1}{4}$.

Zetten we deze punten uit in een cartesiaans assenstelsel en verbinden we de punten door een vloeiende kromme dan krijgen we de grafiek zoals weergegeven in figuur 9.2.

Inderdaad een strikt dalende continue functie in $\mathbb{R}$.



Figuur 9.2: Grafiek van de exponentiële functie met grondtal $\frac{1}{2}$. Merk op: deze grafiek is dezelfde als in Figuur 9.1 maar gespiegeld t.o.v. de Y-as.

Verder lezen we uit de grafiek af dat de y-waarde een zeer groot positief getal wordt wanneer x een zeer groot negatief getal is. Dit betekent: $\lim\limits_{x \to -\infty} f(x) = +\infty$.

Aan de positieve kant nadert de functiewaarde naar 0. Of nog: $\lim\limits_{x \to +\infty} f(x) = 0$. Dit vertaalt zich in een horizontale asymptoot, nl. de X-as, aan de positieve kant.

Voor elke $x \in \mathbb{R}$ geldt steeds $f(x) \in \mathbb{R}_0^+$, dus $\text{bld} f = \mathbb{R}_0^+$.

Aangezien de grafiek van de functie f de X-as nooit bereikt, heeft f geen nulpunten. Het snijpunt met de Y-as is $(0,1)$, immers $f(0) = (\frac{1}{2})^0 = 1$.

Merk op dat de grafiek van de functie $(\frac{1}{2})^x$ de spiegeling is van de grafiek van 2^x t.o.v. de Y-as. Op die manier is het gemakkelijk om de eigenschappen te onthouden.

9.1.2 Eigenschappen

De volgende eigenschappen zijn geldig voor elke exponentiële functie f met grondtal a.

Eigenschappen Stel $f : \mathbb{R} \to \mathbb{R} : x \mapsto f(x) = a^x \quad (a \in \mathbb{R}_0^+ \setminus \{1\})$, dan geldt

- $f(0) = 1$, onafhankelijk het grondtal a.
- $f(1) = a$.
- $\text{dom} f = \mathbb{R}$.

- bld $f = \mathbb{R}_0^+$.
- De functie f is continu in $\mathbb{R}$.
- $a > 1$:
  - $\lim_{x \to +\infty} f(x) = +\infty$
  - $\lim_{x \to -\infty} f(x) = 0$,
    de X-as is een horizontale asymptoot aan de kant van $-\infty$.

  $0 < a < 1$:
  - $\lim_{x \to -\infty} f(x) = +\infty$
  - $\lim_{x \to +\infty} f(x) = 0$,
    de X-as is een horizontale asymptoot aan de kant van $+\infty$.

9.1.3 De natuurlijke exponentiële functie

Er is één bijzondere exponentiële functie die een aparte notatie krijgt. Dit is de exponentiële functie met grondtal e ($e = 2,7182818284\ldots$). Het getal e wordt ook nog het *natuurlijk grondtal* genoemd.

Het getal e speelt een bijzondere rol binnen de wiskunde en werd door Napier voor het eerst ontdekt in 1618. Het voldoet aan de volgende definitie: $e = \lim_{x \to +\infty} (1 + \frac{1}{x})^x$.
Het getal e is een transcendent getal net zoals $\pi \approx 3,14$.

Definitie 9.2 *De **natuurlijke exponentiële functie** met **natuurlijk grondtal** e is*

$$exp : \mathbb{R} \to \mathbb{R} : x \mapsto exp(x) = e^x.$$

9.2 De logaritmische functie

Aangezien een exponentiële functie steeds strikt stijgend of dalend is, is zijn inverse ook een functie. Deze functie wordt de logaritmische functie genoemd.

Definitie 9.3 *De **logaritmische functie met grondtal** a ($a \in \mathbb{R}_0^+ \setminus \{1\}$) is*

$$f : \mathbb{R} \to \mathbb{R} : x \mapsto f(x) = log_a x.$$

Voor $0 < a < 1$ is $\log_a$ strikt dalend.
Voor $a > 1$ is $\log_a$ strikt stijgend.

Voor de exponentiële functie bekeken we als voorbeeld de functie met grondtal $a = 2$ en $a = \frac{1}{2}$. We stellen voor beide functies hun inverse voor, als illustratie van een logaritmische functie.

9.2.1 Voorbeelden

Voorbeeld 1: de logaritmische functie f met grondtal $a = 2$.

$$f : \mathbb{R} \to \mathbb{R} : x \mapsto f(x) = \log_2 x$$

Alvorens de functie te tekenen, berekenen we eerst een aantal puntenkoppels die tot de functie f behoren. De resultaten verzamelen we opnieuw in een tabel.
Om duidelijk te illustreren dat de functie met voorschrift $y = 2^x$ de inverse is van de functie $y = \log_2 x$, plaatsen we de tabel, die we reeds opgesteld hadden voor 2^x, naast de nieuwe tabel.

x	2^x	x	$\log_2 x$
-2	$\frac{1}{4}$	$\frac{1}{4}$	$\log_2 \frac{1}{4} = \log_2 2^{-2} = -2$
-1	$\frac{1}{2}$	$\frac{1}{2}$	$\log_2 \frac{1}{2} = \log_2 2^{-1} = -1$
0	1	1	$\log_2 1 = \log_2 2^0 = 0$
1	2	2	$\log_2 2 = \log_2 2^1 = 1$
2	4	4	$\log_2 4 = \log_2 2^2 = 2$

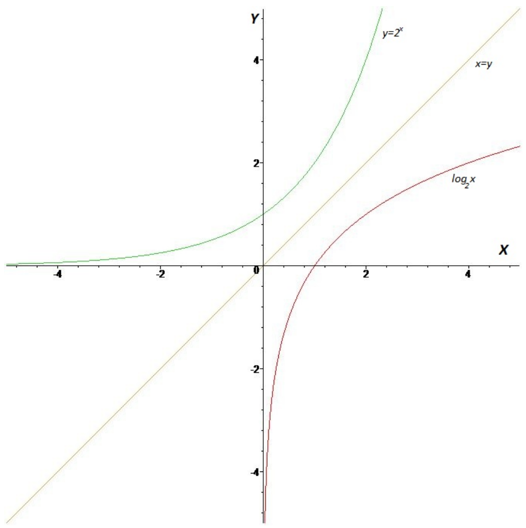
We tekenen de berekende punten in een cartesiaans assenstelsel en verbinden ze door een vloeiende kromme. Dit resulteert in de grafiek zoals weergegeven in figuur 9.3. Op deze tekening wordt eveneens de grafiek van de exponentiële functie met grondtal 2 afgebeeld samen met de rechte $x = y$. Indien de grafiek van 2^x gespiegeld wordt t.o.v. deze rechte $x = y$ verkrijgen we de grafiek van $\log_2 x$.

Verder leert de grafiek ons dat de functie $f(x) = \log_2 x$ enkel gedefinieerd is voor positieve x-waarden. Dit betekent dat het domein van de functie beperkt wordt tot $\mathbb{R}_0^+$, dit is nl. het beeld van de exponentiële functie. Binnen dit domein is deze functie continu.

Op de Y-as worden alle waarden bereikt dus het beeld van de functie f is $\mathbb{R}$, dit is net het domein van de exponentiële functie.

Wat met de punten die aan de *rand* liggen van het domein, meer in het bijzonder 0 en $+\infty$?

- Aan de kant van $+\infty$:
  Hoe groter x wordt hoe groter $\log_2 x$ wordt, dus $\lim_{x \to +\infty} f(x) = +\infty$.
- In een omgeving van 0:
  Hoe dichter x naar 0 nadert, en dit van de rechterkant, hoe meer de functiewaarde naar $-\infty$ nadert. Of nog de **rechterlimiet** $\lim_{x \to_> 0} f(x) = -\infty$. De functie f is niet gedefinieerd voor waarden kleiner dan 0 bijgevolg bestaat dé limiet voor x naderend naar 0 niet. In dit geval kunnen we van een **verticale asymptoot** spreken. De grafiek nadert naar de verticale rechte met vergelijking $x = 0$, d.i. de Y-as.



Figuur 9.3: Grafiek van de logaritmische functie met grondtal 2 en haar inverse.

De gegeven logaritmische functie heeft geen snijpunten met de Y-as, wel een snijpunt met de X-as: $f(1) = \log_2 1 = \log_2 2^0 = 0$. Dus $x = 1$ is het enige nulpunt van de functie f.

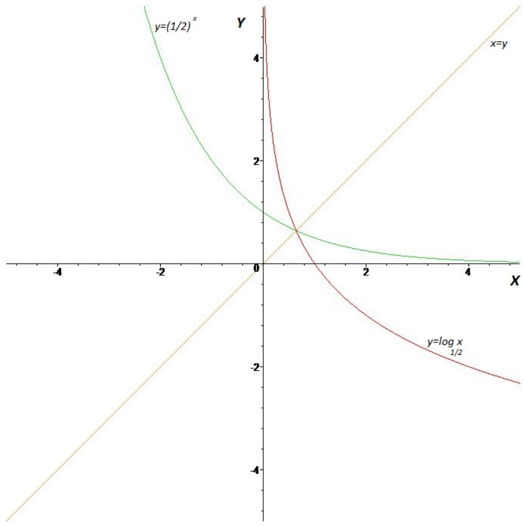
Voorbeeld 2: de logaritmische functie f met grondtal $a = \frac{1}{2}$.

$$f : \mathbb{R} \to \mathbb{R} : x \mapsto f(x) = \log_{\frac{1}{2}} x$$

We plaatsen opnieuw een aantal puntenkoppels van de functie zelf en haar inverse naast elkaar in twee tabellen.

x	$(\frac{1}{2})^x$	x	$\log_{\frac{1}{2}} x$
-2	4	4	$\log_{\frac{1}{2}} 4 = \log_{\frac{1}{2}} (\frac{1}{2})^{-2} = -2$
-1	2	2	$\log_{\frac{1}{2}} 2 = \log_{\frac{1}{2}} (\frac{1}{2})^{-1} = -1$
0	1	1	$\log_{\frac{1}{2}} 1 = \log_{\frac{1}{2}} (\frac{1}{2})^{0} = 0$
1	$\frac{1}{2}$	$\frac{1}{2}$	$\log_{\frac{1}{2}} \frac{1}{2} = \log_{\frac{1}{2}} (\frac{1}{2})^{1} = 1$
2	$\frac{1}{4}$	$\frac{1}{4}$	$\log_{\frac{1}{2}} \frac{1}{4} = \log_{\frac{1}{2}} (\frac{1}{2})^{2} = 2$

We tekenen deze punten in een cartesiaans assenstelsel en verbinden ze door een vloeiende kromme. Dit resulteert in de grafiek zoals weergegeven in figuur 9.4.



Figuur 9.4: Grafiek van de logaritmische functie met grondtal $\frac{1}{2}$ en haar inverse.

Op de grafiek zien we dat de logaritmische functie enkel gedefinieerd is voor strikt positieve x-waarden. Op de Y-as worden alle waarden bereikt. Dus $\operatorname{dom} f = \mathbb{R}_0^+$, $\operatorname{bld} f = \mathbb{R}$.

Verder kunnen we besluiten:

- aan de kant van $+\infty$:
  $\lim\limits_{x \to +\infty} f(x) = -\infty$ (strikt dalende functie).
- in de omgeving van 0:
  de grafiek nadert naar de Y-as met $\lim\limits_{x \to_{>} 0} f(x) = +\infty$.
  De Y-as is dus een verticale asymptoot.

Analoog aan het voorgaande voorbeeld heeft ook deze logaritmische functie geen snijpunten met de Y-as, wel een snijpunt met de X-as: $f(1) = \log_{\frac{1}{2}} 1 = \log_{\frac{1}{2}} (\frac{1}{2})^0 = 0$. Dus $x = 1$ is opnieuw het enige nulpunt van de functie f.

9.2.2 Eigenschappen

De volgende eigenschappen zijn geldig voor elke logaritmische functie f.

Eigenschappen Stel $f : \mathbb{R} \to \mathbb{R} : x \mapsto f(x) = \log_a x \quad (a \in \mathbb{R}_0^+ \setminus \{1\})$ dan geldt:

- $f(1) = 0$, onafhankelijk het grondtal.

- $f(a) = 1$.
- $\text{dom}\, f = \mathbb{R}_0^+$.
- $\text{bld}\, f = \mathbb{R}$.
- De functie f is continu in $\mathbb{R}_0^+$.
- Voor $a > 1$:
  - $\lim\limits_{x \to +\infty} f(x) = +\infty$ (stijgende functie).
  - $\lim\limits_{x \to_> 0} f(x) = -\infty$. De Y-as is verticale asymptoot.

  Voor $0 < a < 1$:
  - $\lim\limits_{x \to +\infty} f(x) = -\infty$ (dalende functie).
  - $\lim\limits_{x \to_> 0} f(x) = +\infty$. De Y-as is verticale asymptoot.

9.2.3 Bijzondere logaritmen

De inverse van de *natuurlijke exponentiële* functie wordt de *Neperiaanse of natuurlijke logaritmische* functie genoemd.
Naast dit natuurlijk logaritme wordt er nog een tweede bijzondere logaritmische functie gedefinieerd, dit is de logaritmische functie met grondtal 10.
In de informatica is het grondtal 2 ook belangrijk. Soms krijgt de logaritmische functie met grondtal 2 een aparte notatie.

Definitie 9.4

- *De **Briggse logaritmische functie** is de logaritmische functie met grondtal 10*

$$log : \mathbb{R}_0^+ \to \mathbb{R} : x \mapsto y = log\, x.$$

- *De **Neperiaanse of natuurlijke logaritmische** functie is de logaritmische functie met grondtal e*

$$ln : \mathbb{R}_0^+ \to \mathbb{R} : x \mapsto y = ln\, x.$$

- *De logaritmische functie met grondtal* 2 *wordt genoteerd als* $lg\, x$.

$$lg : \mathbb{R}_0^+ \to \mathbb{R} : x \mapsto y = lg\, x.$$

Voorbeelden

- $\lg(8) = \log_2 8 = \log_2 2^3 = 3$
- $\lg(2) = \log_2 2 = \log_2 2^1 = 1$
- $\lg(1) = \log_2 1 = \log_2 2^0 = 0$
- $\lg(\frac{1}{8}) = \log_2 \frac{1}{8} = \log_2 2^{-3} = -3$
- $\lg(\frac{1}{2}) = \log_2 \frac{1}{2} = \log_2 2^{-1} = -1$
- $\ln(e) = \log_e e^1 = 1$
- $\log(1000) = \log_{10} 10^3 = 3$

9.2.4 Rekenregels

Gelet op het feit dat de logaritmische functie de inverse is van de exponentiële functie gelden de volgende rekenregels voor de logaritmische functie.

Eigenschap 9.1 *Stel $a, b \in \mathbb{R}_0^+ \setminus \{1\}$, dan geldt:*

$$\begin{array}{rcl} log_a(x \cdot y) & = & log_a x + log_a y \\ log_a \frac{x}{y} & = & log_a x - log_a y \\ log_a(x^p) & = & p \cdot log_a x \quad (met\ p \in \mathbb{Q}) \\ log_a a & = & 1 \\ log_a x & = & \dfrac{log_b x}{log_b a}. \end{array}$$

Voorbeelden

- $\log_4 2 + \log_4 8 = \log_4(2 \cdot 8) = \log_4 16 = \log_4 4^2 = 2$
- $\log_3 36 - \log_3 4 = \log_3 \frac{36}{4} = \log_3 9 = \log_3 3^2 = 2$
- $2 \cdot \log_4 8 = \log_4 8^2 = \log_4 64 = \log_4 4^3 = 3$
- $\log_7 7 = 1$
- $\log_{25} 125 = \dfrac{\log_5 125}{\log_5 25} = \dfrac{\log_5 5^3}{\log_5 5^2} = \dfrac{3}{2}$

9.3 Oefeningen

1. Bepaal het domein en beeld van f. Geef de nulpunten, het snijpunt met de Y-as en eventuele asymptoten van f. Teken ten slotte de grafiek van f.

   (a) $f : \mathbb{R} \to \mathbb{R} : x \mapsto y = \log_3 x$

   (b) $f : \mathbb{R} \to \mathbb{R} : x \mapsto y = 2 + 3^x$

   (c) $f : \mathbb{R} \to \mathbb{R} : x \mapsto y = -2\log_3 x$

2. Stel
$$f : \mathbb{R} \to \mathbb{R} : x \mapsto \ln(\exp(x))$$
$$g : \mathbb{R} \to \mathbb{R} : x \mapsto \exp(\ln(x)).$$
   Wat is het verschil tussen de functie f en de functie g?

3. Je speelt het spel Angry Birds. Je moet door boze vogels te lanceren zoveel mogelijk groene varkens raken. Je krijgt hiervoor drie pogingen. Er bevinden zich groene varkens op de punten met coördinaten: $(1,4)$, $(2,4)$, $(2,5)$, $(3,1)$, $(3,3)$, $(3,4)$, $(3,5)$, $(4,2)$, $(4,4)$.
   Je mag drie vogels lanceren: een bruine, rode en blauwe vogel.

   - De bruine vogel start in het punt met coördinaten $(1,3)$ en vliegt volgens de grafiek van de functie f met $f(x) = 2^x + 1$.
   - De rode vogel start in het punt met coördinaten $(2,0)$ en vliegt volgens de grafiek van de functie g met $g(x) = \lg(x-1)$.
   - De blauwe vogel start in het punt met coördinaten $(2,3)$ en vliegt volgens de grafiek van de functie h met $h(x) = -x^2 + 6x - 5$.

   Beantwoord de volgende vragen:

   a) Maak een tekening van de gegeven situatie. Duid eveneens de vlucht van elke vogel op jouw tekening aan.

   b) Raakt elke vogel een varken? Zo ja, welk varken wordt er geraakt door welke vogel?
   Motiveer jouw antwoord a.d.h.v. een analytische berekening. Verifieer vervolgens of jouw berekeningen overeenstemmen met jouw tekening.

   c) De varkens met coördinaten $(2,4)$ en $(4,2)$ zijn nog niet geraakt. Je krijgt twee extra vogels om specifiek deze varkens uit te schakelen. Beide vogels vertrekken van het punt met coördinaten $(1,2)$. Ze vliegen allebei volgens een rechte lijn.
   Geef voor beide vogels de vergelijking van de functie die hun vlucht volgt.

9.4 Extra oefeningen

1. Los op naar $x \in \mathbb{R}$.

   a) $5^x = \frac{1}{5}$
   b) $7^x = \sqrt[4]{49}$
   c) $2^x = 8$
   d) $4^x = \dfrac{1}{16}$

2. Bereken

   a) $e^3 = \ldots$
   b) $e^{-\frac{2}{3}} = \ldots$
   c) $5^6 = \ldots$
   d) $\log_5 25 = \ldots$
   e) $\log_2 \dfrac{1}{16} = \ldots$
   f) $\log_2 40 = \ldots$
   g) $\log_4 8 + \log_4 2 = \ldots$
   h) $\ln 25 = \ldots$
   i) $\log \sqrt{10} = \ldots$
   j) $\log_2 3 - \log_2 6 = \ldots$

3. Los op naar $x \in \mathbb{R}$.

   a) $\log_2 x = \log_2 5 + \log_2 \frac{1}{3}$
   b) $\log_3(2x + 1) = 1 + \log_3 2$
   c) $\log_3 x = 1$
   d) $\log_5 x = 0$
   e) $\log_8 x = -\frac{2}{3}$
   f) $\log_3 x = 3 \cdot \log_3 4$
   g) $\log x^2 = \log 2^5 - \log 2$
   h) $\log_2 x = 3 \cdot \log_2 3 - \frac{1}{3} \cdot \log_2 27$

4. Geef het domein, het beeld, eventuele nulpunten en snijpunten met de Y-as van de gegeven functies. Noteer eventuele asymptoten. Schets de grafiek.

   a) $f : \mathbb{R} \to \mathbb{R} : x \mapsto f(x) = \exp(x^2)$
   b) $g : \mathbb{R} \to \mathbb{R} : x \mapsto g(x) = \exp(x^2 - 4)$

5. Geef het domein, het beeld, eventuele nulpunten en snijpunten met de Y-as van de gegeven functies. Noteer eventuele asymptoten. Schets de grafiek.

   a) $f : \mathbb{R} \to \mathbb{R} : x \mapsto f(x) = \ln(x^2)$

   b) $g : \mathbb{R} \to \mathbb{R} : x \mapsto g(x) = \ln(x^2 - 4)$

9.5 Oplossingen

1. a) $5^x = \dfrac{1}{5} \Leftrightarrow 5^x = 5^{-1} \Leftrightarrow x = -1$

   b) $7^x = \sqrt[4]{49} \Leftrightarrow 7^x = 7^{\frac{2}{4}} \Leftrightarrow 7^x = 7^{\frac{1}{2}} \Leftrightarrow x = \frac{1}{2}$

   c) $2^x = 8 \Leftrightarrow 2^x = 2^3 \Leftrightarrow x = 3$

   d) $4^x = \dfrac{1}{16} \Leftrightarrow 4^x = \dfrac{1}{4^2} \Leftrightarrow 4^x = 4^{-2} \Leftrightarrow x = -2$

2. a) $\mathrm{e}^3 \approx 20,086$ (met ZRM)

   b) $\mathrm{e}^{-\frac{2}{3}} \approx 0,513$ (met ZRM)

   c) $5^6 = 15\,625$ (met ZRM)

   d) $\log_5 25 = \log_5 5^2 = 2$

   e) $\log_2 \dfrac{1}{16} = \log_2 2^{-4} = -4$

   f) $\log_2 40 = \dfrac{\ln 40}{\ln 2} \approx 5,322$ (met ZRM)

   g) $\log_4 8 + \log_4 2 = \log_4(8 \cdot 2) = \log_4 16 = \log_4 4^2 = 2$

   h) $\ln 25 \approx 3,219$ (met ZRM)

   i) $\log \sqrt{10} = \log 10^{\frac{1}{2}} = \frac{1}{2}$

   j) $\log_2 3 - \log_2 6 = \log_2 \dfrac{3}{6} = \log_2 \dfrac{1}{2} = \log_2 2^{-1} = -1$

3. a)

$$\begin{array}{rcl} \log_2 x & = & \log_2 5 + \log_2 \dfrac{1}{3} \\ & \Updownarrow & \\ \log_2 x & = & \log_2 \dfrac{5}{3} \\ & \Updownarrow & \\ x & = & \dfrac{5}{3} \end{array}$$

b)

$$\begin{array}{rcl} \log_3(2x+1) & = & 1+\log_3 2 \\ & \Updownarrow & \\ \log_3(2x+1) & = & \log_3 3+\log_3 2 \\ & \Updownarrow & \\ \log_3(2x+1) & = & \log_3 6 \\ & \Updownarrow & \\ 2x+1 & = & 6 \\ & \Updownarrow & \\ x & = & \frac{5}{2} \end{array}$$

c)

$$\begin{array}{rcl} \log_3 x & = & 1 \\ & \Updownarrow & \\ \log_3 x & = & \log_3 3^1 \\ & \Updownarrow & \\ x & = & 3 \end{array}$$

d)

$$\begin{array}{rcl} \log_5 x & = & 0 \\ & \Updownarrow & \\ \log_5 x & = & \log_5 5^0 \\ & \Updownarrow & \\ x & = & 1 \end{array}$$

e)

$$\begin{array}{rcl} \log_8 x & = & -\frac{2}{3} \\ & \Updownarrow & \\ \log_8 x & = & \log_8 8^{-\frac{2}{3}} \\ & \Updownarrow & \\ x & = & \frac{1}{\sqrt[3]{8^2}} \\ & \Updownarrow & \\ x & = & \frac{1}{\sqrt[3]{64}} \\ & \Updownarrow & \\ x & = & \frac{1}{4} \end{array}$$

f)

$$
\begin{array}{rcl}
\log_3 x & = & 3 \cdot \log_3 4 \\
& \Updownarrow & \\
\log_3 x & = & \log_3 4^3 \\
& \Updownarrow & \\
x & = & 4^3 \\
& \Updownarrow & \\
x & = & 64
\end{array}
$$

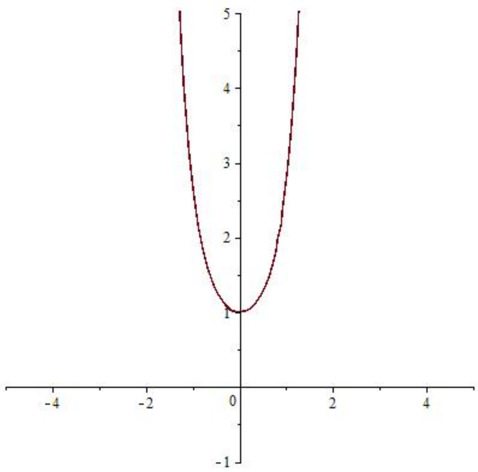
g)

$$\begin{array}{rcl}
\log x^2 & = & \log 2^5 - \log 2 \\
& \Updownarrow & \\
\log x^2 & = & \log \dfrac{2^5}{2} \\
& \Updownarrow & \\
x^2 & = & 2^4 \\
& \Updownarrow & \\
x & = & \pm\sqrt{2^4} \\
& \Updownarrow & \\
x = 4 & \text{of} & x = -4
\end{array}$$

h)

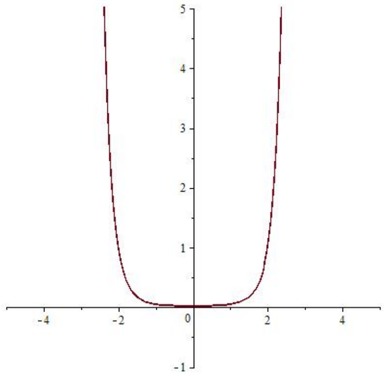
$$\begin{array}{rcl}
\log_2 x & = & 3 \cdot \log_2 3 - \frac{1}{3} \cdot \log_2 27 \\
& \Updownarrow & \\
\log_2 x & = & \log_2 3^3 - \log_2 27^{\frac{1}{3}} \\
& \Updownarrow & \\
\log_2 x & = & \log_2 \frac{27}{3} \\
& \Updownarrow & \\
x & = & 9
\end{array}$$

4. a)
- $\operatorname{dom} f = \mathbb{R}$.
- $\operatorname{bld} f = [1, +\infty[$.
- Er zijn geen nulpunten.
- Snijpunt Y-as: $(0, 1)$ (Verkl: $f(0) = e^0 = 1$).
- Er zijn geen asymptoten.
- Grafiek:



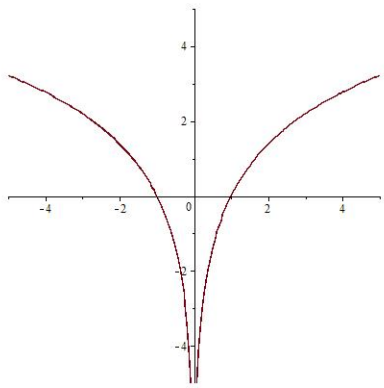
Figuur 9.5: De grafiek van de functie $f : \mathbb{R} \to \mathbb{R} : x \mapsto f(x) = \exp(x^2)$.

b)
- dom $g = \mathbb{R}$.
- bld $g = [\frac{1}{e^4}, +\infty[$.
- Er zijn geen nulpunten.
- Snijpunt Y-as: $(0, e^{-4})$.
- Er zijn geen asymptoten.
- Grafiek:



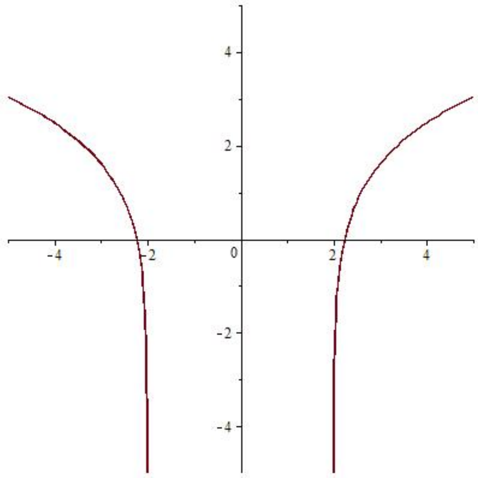
Figuur 9.6: De grafiek van de functie $g : \mathbb{R} \to \mathbb{R} : x \mapsto g(x) = \exp(x^2 - 4)$.

5. a)
- dom $f = \mathbb{R}_0$.
- bld $f = \mathbb{R}$.
- Nulpunten: $x \in \{-1, 1\}$.
- Geen snijpunten met Y-as.
- Asymptoten: er is één verticale asymptoot, nl. de Y-as.
- Grafiek:



Figuur 9.7: De grafiek van de functie $f : \mathbb{R} \to \mathbb{R} : x \mapsto f(x) = \ln(x^2)$.

b)
- dom $g = \mathbb{R} \setminus [-2, 2]$.
- bld $g = \mathbb{R}$.
- Nulpunten: $x \in \{-\sqrt{5}, \sqrt{5}\}$.
- Er zijn geen snijpunten met de Y-as.
- Asymptoten: er zijn twee verticale asymptoten, nl. de rechten met vergelijking $x = -2$ en $x = 2$.
- Grafiek:



Figuur 9.8: De grafiek van de functie $g : \mathbb{R} \to \mathbb{R} : x \mapsto g(x) = \ln(x^2 - 4)$.

HOOFDSTUK **10**

Bijzondere functies

Als afsluiting van het deel Analyse bespreken we drie bijzondere functies die vaak gebruikt worden in de informatica.

Als eerste bijzondere functie behandelen we de *absolute waarde* functie. De twee volgende functies zijn de *floor* functie en de *ceiling* functie.

De *floor* en de *ceiling* functie zijn *discontinue functies*.
Een discontinue functie is een functie waarvoor de grafiek niet kan getekend worden in één vloeiende beweging. De grafiek vertoont minstens één sprong.

10.1 De absolute waarde functie

De *absolute waarde functie* is de functie die een gegeven getal x afbeeldt op x zelf als x een positief getal is, op $-x$ als x een negatief getal is.

Definitie 10.1 *De **absolute waarde functie** is de functie met als functievoorschrift*

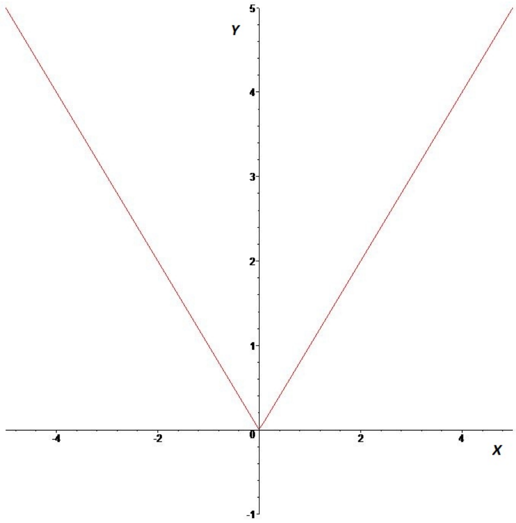
$$abs : \mathbb{R} \to \mathbb{R} : x \mapsto \sqrt{x^2} = \begin{cases} x & \text{als } x \geq 0 \\ -x & \text{als } x < 0. \end{cases}$$

De grafiek van de functie abs wordt weergegeven in Figuur 10.1.

Op deze tekening zien we dat de grafiek van abs een knik vertoont in de oorsprong, het punt (0,0).

Verder geeft de grafiek aan dat enkel positieve y-waarden bereikt worden: bld(abs)= $\mathbb{R}^+$.

Het domein van de functie is $\mathbb{R}$: dom(abs)= $\mathbb{R}$.



Figuur 10.1: Grafiek van de functie abs.

Voorbeelden

- $\text{abs}(7) = 7$
- $\text{abs}(-13) = 13$

10.2 *Floor* en *Ceiling* functie

Zowel de floor als de ceiling functie zijn discontinue functies. Voor beide functies vertoont de grafiek meerdere sprongen.

10.2.1 De functie *Floor*

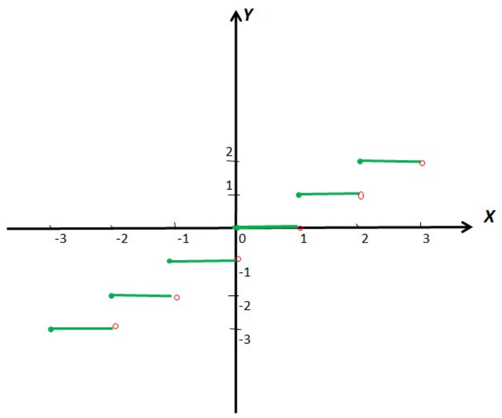
De *floor*-functie beeldt een reëel getal x af op het dichtstbij gelegen geheel getal z, met $z \leq x$. De floor functie rondt m.a.w. af naar beneden.

Definitie 10.2 *De **floor**-functie is de functie met functievoorschrift:*

$$\mathit{floor} : \mathbb{R} \to \mathbb{Z} : x \mapsto \mathit{floor}(x) = z \text{ met } z \text{ het grootste geheel getal zodat } z \leq x.$$

Deze functie is gedefinieerd voor alle reële getallen x, maar het beeld z kan uiteraard enkel een geheel getal zijn. Dus dom(floor)= $\mathbb{R}$ en bld(floor)= $\mathbb{Z}$.

De grafiek van de functie wordt gegeven in Figuur 10.2.



Figuur 10.2: Grafiek van de functie floor.

Voorbeelden

- $\text{floor}(2,77) = 2$
- $\text{floor}(3,03) = 3$
- $\text{floor}(5,0) = 5$
- $\text{floor}(-0,5) = -1$
- $\text{floor}(-4,9) = -5$

10.2.2 De functie *Ceiling*

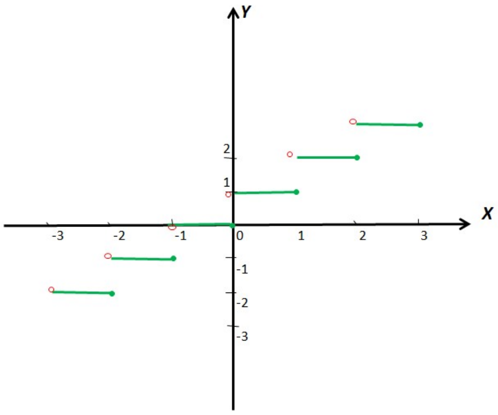
Analoog aan de floor functie beeldt de *ceiling*-functie een reëel getal x af op het dichtstbij gelegen geheel getal z, maar deze keer moet $z \geq x$. De ceiling functie rondt m.a.w. af naar boven.

Definitie 10.3 *De **ceiling**-functie is de functie met functievoorschrift:*

$$ceiling : \mathbb{R} \to \mathbb{Z} : x \mapsto ceiling(x) = z \textit{ met } z \textit{ het kleinste geheel getal zodat } z \geq x.$$

De grafiek van de functie wordt gegeven in Figuur 10.3.

Opnieuw geldt: dom(ceiling)= $\mathbb{R}$, bld(ceiling)= $\mathbb{Z}$.



Figuur 10.3: Grafiek van de functie ceiling.

Voorbeelden

- $\text{ceiling}(2,77) = 3$
- $\text{ceiling}(3,03) = 4$
- $\text{ceiling}(5,0) = 5$
- $\text{ceiling}(-0,5) = 0$
- $\text{ceiling}(-4,9) = -4$

10.3 Oefeningen

1. Stel $f : \mathbb{R} \to \mathbb{R} : x \mapsto \text{abs}(x+1)$.
   a) Geef het domein en beeld van f.
   b) Bepaal alle nulpunten van f.
   c) Schets de grafiek van f.
2. Stel
$$f : \mathbb{R} \to \mathbb{R} : x \mapsto \text{floor(ceiling}(x - \tfrac{1}{2}))$$
$$g : \mathbb{R} \to \mathbb{R} : x \mapsto \text{floor(ceiling}(x) - \tfrac{1}{2})$$
$$h : \mathbb{R} \to \mathbb{R} : x \mapsto \text{ceiling(floor}(x + \tfrac{1}{2})).$$
   Geef voor elk van de gegeven functies het domein en het beeld. Bepaal voor elke functie de nulpunten. Teken de grafiek van de functies f, g en h.

10.4 Extra oefeningen

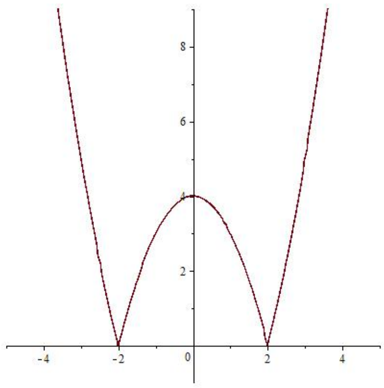
1. Stel $f : \mathbb{R} \to \mathbb{R} : x \mapsto f(x) = \text{abs}(x^2 - 4)$.
   Bepaal het domein en beeld van f. Noteer alle nulpunten en het snijpunt met de Y-as. Schets de grafiek van f.

2. Stel $f : \mathbb{R} \to \mathbb{R} : x \mapsto f(x) = \text{ceiling}(\text{floor}(x - 4) + 4)$.
   Bepaal het domein en beeld van f. Noteer alle nulpunten en het snijpunt met de Y-as. Schets de grafiek van f.

3. Stel $f : \mathbb{R} \to \mathbb{R} : x \mapsto f(x) = \log_2(2^{\text{floor}(x)})$.
   Bepaal het domein en beeld van f. Noteer alle nulpunten en het snijpunt met de Y-as. Schets de grafiek van f.

10.5 Oplossingen

1.
   - $\text{dom} f = \mathbb{R}$.
   - $\text{bld} f = \mathbb{R}^+$.
   - Nulpunten: $x \in \{-2, 2\}$.
   - Snijpunt Y-as: $(0, 4)$.
   - Grafiek:



Figuur 10.4: De grafiek van de functie $f : \mathbb{R} \to \mathbb{R} : x \mapsto f(x) = \text{abs}(x^2 - 4)$.

2.
   - $\text{dom} f = \mathbb{R}$.
   - $\text{bld} f = \mathbb{Z}$.
   - Nulpunten: $x \in [0, 1[$.
   - Snijpunt Y-as: $(0, 0)$.

- Grafiek:
  de grafiek van de functie komt overeen met de grafiek van de floor functie aangezien $f(x) = \text{ceiling}(\text{floor}(x - 4) + 4) = \text{floor}(x)$.

3.
- dom $f = \mathbb{R}$.
- bld $f = \mathbb{Z}$.
- Nulpunten: $x \in [0, 1[$.
- Snijpunt Y-as: $(0, 0)$.
- Grafiek:
  de grafiek van de functie komt overeen met de grafiek van de floor functie aangezien $f(x) = \log_2(2^{\text{floor}(x)}) = \text{floor}(x)$.

HOOFDSTUK **11**

Eindige velden

In dit hoofdstuk zullen we ons toespitsen op de wiskundige theorie van *eindige velden*.

Eindige velden hebben vele praktische toepassingen. Twee voorbeelden hiervan zijn codeertheorie en cryptografie.
Codeertheorie wordt verder besproken in de cursus computerarchitectuur. Cryptografie zal aan bod komen binnen netwerken.

11.1 Definities en eigenschappen

Alvorens de definitie van een *eindig veld* te geven, herhalen we de definitie van een *veld*.

Definitie 11.1 *Een verzameling F waarop twee binaire operatoren, $+$ (optelling) en $\cdot$ (vermenigvuldiging), gedefinieerd zijn en waartoe zeker twee constante elementen, 0 en 1, behoren is een* ***veld*** *wanneer de volgende eigenschappen voldaan zijn: Stel $a, b, c \in F$*

- *F is gesloten voor $+$ en $\cdot$: $a + b \in F$ en $a \cdot b \in F$.*
- *commutatief: $a + b = b + a$ en $a \cdot b = b \cdot a$.*
- *associatief: $(a + b) + c = a + (b + c)$ en $(a \cdot b) \cdot c = a \cdot (b \cdot c)$.*
- *distributief: $a \cdot (b + c) = a \cdot b + a \cdot c$.*
- *neutraal element: $a + 0 = a$ en $a \cdot 1 = a$.*
- *invers element voor +: er bestaat een element $-a \in F$ zodat $a + (-a) = 0$.*
- *invers element voor $\cdot$: voor elke $a \in F$, met $a \neq 0$ bestaat er een element a^{-1} zodat $a \cdot a^{-1} = 1$.*

Voorbeeld
$\mathbb{R}, +, \cdot$ is een veld.
$\mathbb{Z}, +, \cdot$ is geen veld: niet elk element heeft een invers element voor de vermenigvuldiging.

Eigenschap 11.1 *Voor een willekeurig veld F met $a, b \in F$ geldt*

- *het element* 0 *is het opslorpend element voor de vermenigvuldiging:* $a \cdot 0 = 0$.
- *een veld F heeft geen* ***nuldelers****: als $a \cdot b = 0$ dan geldt $a = 0$ of $b = 0$.*

De tweede eigenschap kan heel nuttig zijn om na te gaan of een bepaalde structuur al dan niet voldoet aan de regels van een veld.

Notatie
Voortaan zullen we $a \cdot b$ eveneens noteren als ab.

Definitie 11.2 *Een* ***eindig veld*** *is een veld waarvoor de verzameling van elementen eindig is. Het aantal elementen van deze verzameling is de* ***orde*** *van het veld.*

Stelling 11.1 *Er bestaat een veld van de orde q als en slechts als q de macht van een priemgetal p is ($q = p^h$, met $h \in \mathbb{N}_0$).*
Twee velden van de orde q met $q = p^h$ zijn isomorf. Een veld van de orde q noemen we een ***Galois veld van de orde*** *q,* $\mathrm{GF}(q)$.

11.2 Het eindig veld $\mathbb{Z}_p$

De verzameling $\mathbb{Z}_m$, met $m \in \mathbb{N}_0$, stelt de verzameling voor met als elementen de eerste m natuurlijke getallen:

$$\mathbb{Z}_m = \{0, 1, \ldots, m-1\}.$$

De verzameling $\mathbb{Z}_m$ bevat m verschillende elementen: $\#\mathbb{Z}_m = m$. Dus $\mathbb{Z}_m$ is een eindige verzameling.

Indien we wensen te rekenen met de elementen van $\mathbb{Z}_m$ dan zal de optelling en de vermenigvuldiging zoals gedefinieerd voor $\mathbb{Z}$ niet volstaan, aangezien niet elk resultaat opnieuw tot $\mathbb{Z}_m$ zal behoren.

Voorbeeld Beschouw de verzameling $\mathbb{Z}_8 = \{0, 1, 2, 3, 4, 5, 6, 7\}$.
Er geldt $3 + 4 = 7 \in \mathbb{Z}_8$.
Maar $3 \cdot 4 = 12 \notin \mathbb{Z}_8$.

Kunnen we een optelling en vermenigvuldiging definiëren op $\mathbb{Z}_m$ zodat $\mathbb{Z}_m, +, \cdot$ de structuur van een veld krijgt?

De definitie van + en $\cdot$ maakt gebruik van de *modulo* bewerking. We definiëren eerst deze bewerking.

Definitie 11.3 *Stel $a, b \in \mathbb{Z}$. Dan is* ***a congruent met** b **modulo** m als en slechts als de deling van a en van b door m dezelfde rest oplevert.*
We noteren: $a \equiv b \pmod m$, met $a \bmod m =$ de positieve rest na deling door m.

Voorbeelden

$2 \equiv 23 \pmod 7$	$17 \equiv 2 \pmod 5$	$23 \equiv 5 \pmod 9$
$16 \equiv 0 \pmod 4$	$16 \equiv 0 \pmod 8$	$16 \not\equiv 0 \pmod 3$

Eigenschap 11.2 *Stel $a \equiv a' \pmod m$ en $b \equiv b' \pmod m$, dan geldt*

- $a + b \equiv a' + b' \pmod m$.
- $a \cdot b \equiv a' \cdot b' \pmod m$.

Voorbeeld $13 \cdot 19 \equiv 3 \cdot 4 \equiv 2 \pmod 5$

Met behulp van de modulo bewerking is het mogelijk een optelling en vermenigvuldiging te definiëren op de elementen van $\mathbb{Z}_m$ zodat deze beide bewerkingen inwendig zijn. Als de bewerking inwendig is, zal het resultaat van die bewerking opnieuw tot $\mathbb{Z}_m$ behoren.

Definitie 11.4 *Stel $a, b \in \mathbb{Z}_m$. In $\mathbb{Z}_m$ worden de bewerkingen $+$ en $\cdot$ als volgt gedefinieerd:*

$$\begin{aligned} a + b &\equiv a + b \pmod m \\ \text{en} \\ a \cdot b &\equiv a \cdot b \pmod m. \end{aligned}$$

Voorbeeld Beschouw de verzameling $\mathbb{Z}_{13}$, dan geldt

- $8 + 5 \equiv 0 \pmod{13}$.
- $9 + 7 \equiv 3 \pmod{13}$.
- $3 \cdot 5 \equiv 2 \pmod{13}$.
- $3 \cdot 3 \equiv 9 \pmod{13}$.

Met deze definitie voor de optelling is het resultaat van een som van twee elementen van $\mathbb{Z}_m$ opnieuw een element van $\mathbb{Z}_m$. Analoog voor het product. Hernemen we het voorbeeld voor $\mathbb{Z}_8$.

Voorbeeld Beschouw de verzameling $\mathbb{Z}_8 = \{0, 1, 2, 3, 4, 5, 6, 7\}$, er geldt

- $3 + 4 \equiv 7 \pmod 8 \in \mathbb{Z}_8$.
- $3 \cdot 4 \equiv 12 \equiv 4 \pmod 8 \in \mathbb{Z}_8$.

Maar! $2 \cdot 4 \equiv 0 \pmod 8$. Nochtans geen van beide factoren is 0! Gelet op de eigenschap 11.1 kan $\mathbb{Z}_8, +, \cdot$ geen veld zijn.

Stelling 11.2 *$\mathbb{Z}_p, +, \cdot$ is een veld als en slechts als p een priemgetal is.*

11.3 Voorbeelden

Het eindig veld met 2 elementen

Het eindig veld met twee elementen is $\mathrm{GF}(2) = \mathbb{Z}_2 = \{0, 1\}$ met de tabel voor de optelling en vermenigvuldiging

+	0	1
0	0	1
1	1	0

·	0	1
0	0	0
1	0	1

Het eindig veld met 3 elementen

Het eindig veld met drie elementen is $\mathrm{GF}(3) = \mathbb{Z}_3 = \{0, 1, 2\}$ met de tabel voor de optelling en vermenigvuldiging

+	0	1	2
0	0	1	2
1	1	2	0
2	2	0	1

·	0	1	2
0	0	0	0
1	0	1	2
2	0	2	1

Het eindig veld met 4 elementen

Gelet op stelling 11.2 is $\mathbb{Z}_4$ geen veld, 4 is immers geen priemgetal. In $\mathbb{Z}_4$ geldt $2 \cdot 2 \equiv 0 \pmod 4$. Dus 2 is een nuldeler. Hieruit kunnen we onmiddellijk besluiten dat $\mathbb{Z}_4$ geen veld is.
Er geldt $4 = 2^2$ en dus is 4 een macht van een priemgetal. Waaruit volgt dat er wel een eindig veld met 4 elementen moet bestaan. Dit eindig veld met 4 elementen kan als volgt gedefinieerd worden: $\mathrm{GF}(4) = \{0, 1, a, b\}$ met

+	0	1	a	b
0	0	1	a	b
1	1	0	b	a
a	a	b	0	1
b	b	a	1	0

·	0	1	a	b
0	0	0	0	0
1	0	1	a	b
a	0	a	b	1
b	0	b	1	a

Het eindig veld met 5 elementen

Het getal 5 is een priemgetal. Dit betekent dat we opnieuw met $\mathbb{Z}_5$ kunnen werken voor het eindig veld van 5 elementen.
Het eindig veld met 5 elementen is $\text{GF}(5) = \mathbb{Z}_5 = \{0,1,2,3,4\}$ met de optelling en vermenigvuldiging als volgt gedefinieerd in $\mathbb{Z}_5$

$+$	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

$\cdot$	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

Aangezien $\mathbb{Z}_5, +, \cdot$ een veld is, moet elk element een tegengestelde hebben voor de optelling en een invers voor de vermenigvuldiging. Noteren we ook deze twee tabellen:

a	$-a$
0	0
1	4
2	3
3	2
4	1

a	a^{-1}
0	/
1	1
2	3
3	2
4	4

Voor elk element a van $\mathbb{Z}_5$ moet gelden $a + (-a) \equiv 0 \pmod 5$ en voor alle elementen a van $\mathbb{Z}_5$, behalve 0, moet gelden $a \cdot a^{-1} \equiv 1 \pmod 5$.

De verzameling $\mathbb{Z}_6$

Het getal 6 is geen priemgetal dus $\mathbb{Z}_6, +, \cdot$ is geen veld.
De verzameling $\mathbb{Z}_6$ bevat opnieuw nuldelers: $2 \cdot 3 \equiv 0 \pmod 6$. Dit bewijst onmiddellijk het gestelde.
Het is tevens niet mogelijk 6 te schrijven als een macht van een priemgetal. Gelet op stelling 11.1 mogen we besluiten dat er geen enkel eindig veld met 6 elementen bestaat.

Het eindig veld met 11 elementen

Het getal 11 is een priemgetal.
Het eindig veld met 11 elementen is $\text{GF}(11) = \mathbb{Z}_{11} = \{0,1,2,3,4,5,6,7,8,9,10\}$. De optelling en vermenigvuldiging op deze verzameling is analoog aan de voorgaande voorbeelden.

We noteren alle inverse elementen voor de optelling en de vermenigvuldiging in $\mathbb{Z}_{11}$.

a	$-a$
0	0
1	10
2	9
3	8
4	7
5	6
6	5
7	4
8	3
9	2
10	1

a	a^{-1}
0	/
1	1
2	6
3	4
4	3
5	9
6	2
7	8
8	7
9	5
10	10

11.4 Toepassing: de ISBN-code

Elk boek in een boekhandel is voorzien van een Internationaal Standaard BoekNummer (ISBN). In 1966 is het ISBN-nummer ontstaan in Groot-Brittanië. Oorsprokelijk was dit een codewoord bestaande uit 10-digits bepaald door de uitgever.
Bijvoorbeeld:

$$0 - 19 - 859617 - 0.$$

De eerste digit geeft de taal weer, het tweede deel van de code is verbonden met de uitgeverij, het derde deel is een nummer bepaald door de uitgeverij en op het einde volgt er een controlecijfer. Dit controlecijfer laat toe eventuele fouten op te sporen, verbeteren kan echter niet. Meer informatie hierover vind je o.a. op wikipedia[1].

Wiskundig gezien moet de code aan de volgende eisen voldoen:

Definitie 11.5 *Een **ISBN-code** bestaat uit 10 digits:* $\mathbf{x} = x_1x_2x_3\ldots x_{10}$ *met* $x_i \in \{0,1,\ldots,9\}$ *voor* $i = 1,\ldots,9$ *en* $x_{10} \in \{0,1,\ldots,9,X\}$ *waarvoor geldt*

$$\sum_{i=1}^{10} ix_i \equiv 0\,(mod11).$$

Voorbeeld
Hernemen we de code $0 - 19 - 859617 - 0$ dan geldt inderdaad

$$\sum_{i=1}^{10} ix_i \equiv 55 \equiv 0\,(\text{mod}11).$$

[1] http://nl.wikipedia.org/wiki/Internationaal_Standaard_Boeknummer, (15/06/2009).

Een overzichtelijke manier om dit te verifiëren is als volgt:

$$\begin{array}{lllllllllll}
x_i & 0 & 1 & 9 & 8 & 5 & 9 & 6 & 1 & 7 & 0 \\
i & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\
\hline
\sum_{i=1}^{10} ix_i \equiv & 0+ & 2+ & 5+ & 10+ & 3+ & 10+ & 9+ & 8+ & 8+ & 0 \equiv 55 \equiv 0 (\text{mod } 11).
\end{array}$$

Eigenschap 11.3 *Voor een ISBN-code $x_1 \ldots x_{10}$ geldt*

$$x_{10} \equiv \sum_{i=1}^{9} ix_i \, (mod 11).$$

Bewijs:

$$\begin{array}{rcl}
\sum_{i=1}^{10} ix_i \equiv \sum_{i=1}^{9} ix_i + 10x_{10} & \equiv & 0 \quad (\text{mod } 11) \\
& \Updownarrow & \\
\sum_{i=1}^{9} ix_i & \equiv & -10x_{10} \equiv 1x_{10} \quad (\text{mod } 11) \\
& \Updownarrow & \\
\sum_{i=1}^{9} ix_i & \equiv & x_{10} \quad (\text{mod } 11)
\end{array}$$

Voorbeeld
Gelet op de voorgaande eigenschap moet in $0 - 19 - 853803 - x$ de ontbrekende digit x gelijk zijn aan 0 opdat deze code een correcte ISBN-code zou zijn.
Uitwerking:

$$\begin{array}{lllllllllll}
x_i & 0 & 1 & 9 & 8 & 5 & 3 & 8 & 0 & 3 & x \\
i & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\
\hline
\sum_{i=1}^{9} ix_i \equiv & 0+ & 2+ & 5+ & 10+ & 3+ & 7+ & 1+ & 0+ & 5 & \equiv 33 \, (\text{mod } 11)
\end{array}$$

Dus $x \equiv \sum_{i=1}^{9} ix_i \equiv 33 \equiv 0$ (mod 11).

Kan je berekenen hoeveel boeken je maximaal kan voorzien van een ISBN-code[2]?

Na een dertigtal jaren bleken er niet voldoende ISBN-nummers te zijn. Dit leidt tot een aanpassing van de ISBN-code. Sinds 2007 werd de code gewijzigd conform de internationale EAN-standaard (**E**uropese **A**rtikel**N**ummering). Er werd een prefix van drie

[2]Oplossing: 10^9.

extra cijfers voorzien zodat de code nu niet meer uit 10 cijfers maar wel uit 13 cijfers bestaat. Alle bestaande nummers werden uitgebreid met het prefix 978. Voor nieuwe boeken werd het prefix 979 gereserveerd. Hoeveel boeken kan men nu voorzien van een nummer[3]?

De laatste digit van de code is nog steeds een controlecijfer, maar de berekening van dit controlecijfer werd eveneens gewijzigd. Men rekent niet langer in het eindig veld met 11 elementen. Voor het controlecijfer van de nieuwe versie wordt modulo 10 gewerkt en wel als volgt:

- alle digits die op een oneven positie staan worden vermeningvuldigd met 1
- alle digits die op een even positie staan worden vermenigvuldigd met 3

de totale som van al deze termen plus het nieuwe controlecijfer moet een veelvoud van 10 zijn.

Definitie 11.6 *Een ISBN-code opgebouwd volgens de internationale EAN-standaard bestaat uit 13 digits:* $\mathbf{x} = x_1x_2x_3\ldots x_{13}$ *met* $x_i \in \{0, 1, \ldots, 9\}$ *voor* $i = 1, \ldots, 13$ *waarvoor geldt*

$$\sum_{i=0}^{6} x_{2i+1} + 3 \cdot (\sum_{i=1}^{6} x_{2i}) \equiv 0\,(mod 10).$$

Opmerking Alle EAN-codes (nummers horende bij de streepjescode) opgebouwd uit 13 digits voldoen aan deze definitie.

Dit betekent dat bij omzetting van de bestaande ISBN-nummers naar het nieuwe formaat alle controlecijfers opnieuw moesten berekend worden.

Voorbeeld
Hernemen we de oude code $0 - 19 - 859617 - 0$. Willen we deze herschrijven naar de nieuwe vorm dan moeten we het prefix 978 toevoegen:

$$978 - 0 - 19 - 859617 - x_{13}$$

Het controlecijfer moet nu berekend worden volgens de nieuwe regels:

- som van alle digits op oneven posities maal 1: $(9 + 8 + 1 + 8 + 9 + 1 + x_{13}) \cdot 1 = 36 + x_{13}$
- som van alle digits op even posities maal 3: $(7 + 0 + 9 + 5 + 6 + 7) \cdot 3 = 102$.

[3]Oplossing: 2×10^9

De som van beide resultaten is $36 + 102 + x_{13} \equiv 138 + x_{13} \equiv 8 + x_{13} (\mathrm{mod}\, 10)$. Het controlegetal moet zo gekozen worden dat de totale som gelijk is aan 0 als we rekenen mod 10. Dus er moet gelden $8 + x_{13} \equiv 0 (\mathrm{mod}\, 10)$. Dit is voldaan voor $x_{13} = 2$. De correcte code is

$$978 - 0 - 19 - 859617 - 2.$$

11.5 Oefeningen

1. Noteer alle inverse elementen voor de optelling en de vermenigvuldiging in:
   a) $\mathbb{Z}_7$.
   b) $\mathbb{Z}_{13}$.
   c) $\mathbb{Z}_{17}$.
2. Los op naar x in $\mathbb{Z}_{11}$:
   a) $3x + 5 \equiv 0$.
   b) $4x + 6 \equiv 0$.
3. a) Zijn de volgende codes ISBN-codes?
      i. $3 - 411 - 02175 - 6$
      ii. $2 - 85036 - 008 - X$
      iii. $0 - 13165332 - 6$
   b) Zet elke geldige code om naar de standaard sinds 2007.
4. Bereken de weggelaten digit in de ISBN-codes:
   a) $* - 521 - 283987$
   b) $0 - 13 - 1 * 9139 - 9$
5. (Bron [Van Maldeghem, 2004]) Stel je wil een bibliotheek aanleggen met boeken over informatica. Al de boeken uit deze bibliotheek wil je volgens een eigen systeem nummeren. Noemen we dit nummer het SBNI (Standaard Boek Nummer Informatica). Voor de opbouw van het SBNI-nummer werken we in $\mathbb{Z}_5$.

   Het eerste deel van het nummer bestaat uit 4 cijfers, $c_1c_2c_3c_4$ gekozen uit de elementen van $\mathbb{Z}_5$. Na een streepje volgen twee controlecijfers.

   - Het eerste controlecijfer is de som van de eerste vier cijfers, $\sum_{i=1}^{4} c_i$.
   - Het tweede controlecijfer is de som van de eerste vier cijfers telkens vermenigvuldigd met hun positie, $\sum_{i=1}^{4} (i \cdot c_i)$.

   De berekeningen worden uitgevoerd in $\mathbb{Z}_5$.

   a) Ga na of de volgende codes correcte SBNI-nummers zijn:

i. $1043 - 30$
ii. $2134 - 04$
iii. $1963 - 25$

b) Hoeveel boeken kan men nummeren met deze code?

c) In de gegeven codes ontbreken een aantal digits, vul deze in zodat een correct SBNI-nummer ontstaat:

i. $*3*2 - 12$
ii. $**40 - 00$
iii. $24*3 - 0*$

d) Controleer de volgende SBNI-nummers. Indien de codes niet correct zijn, spoor de fout dan op en verbeter ze.

i. $2410 - 23$
ii. $0023 - 14$
iii. $1960 - 11$

11.6 Extra oefeningen

1. (Bron [Van Maldeghem, 2004]) Stel je wil een bibliotheek aanleggen met boeken over informatica. Al de boeken uit deze bibliotheek wil je volgens een eigen systeem nummeren. Noemen we dit nummer het SBNI (Standaard Boek Nummer Informatica).

   Voor de opbouw van het SBNI-nummer werken we in $\mathbb{Z}_7$.

   Het eerste deel van het nummer bestaat uit 6 cijfers, $c_1c_2c_3c_4c_5c_6$ gekozen uit de elementen van $\mathbb{Z}_7$. Na het streepje volgen deze keer vier controlecijfers.

   - Het eerste controlecijfer is de som van de eerste zes cijfers, $\sum_{i=1}^{6} c_i$.
   - Het tweede controlecijfer is de som van de eerste zes cijfers telkens vermenigvuldigd met hun positie, $\sum_{i=1}^{6}(i \cdot c_i)$.
   - Het derde controlecijfer is de som $\sum_{i=1}^{6}(i^2 \cdot c_i)$.
   - Het laatste controlecijfer is de som $\sum_{i=1}^{6}(i^3 \cdot c_i)$.

   De berekeningen worden uitgevoerd in $\mathbb{Z}_7$.

   a) Hoeveel boeken kan men nummeren met deze code?

   b) In de gegeven codes ontbreken een aantal digits, vul deze in zodat een correct SBNI-nummer ontstaat:

   i. $625316 - ****$
   ii. $1*3526 - **3*$
   iii. $4*23*0 - *00*$

iv. $*0***0-5555$

2. Stel $a \equiv 5 \pmod 6$. Bereken:

   a) $a \pmod 3$.

   b) $a \pmod{12}$.

3. Bepaal $m \in \mathbb{N}_0$ zodat de gegeven uitdrukking geldig is. Indien meerdere oplossingen mogelijk zijn, moet de kleinst mogelijke waarde van m gegeven worden.

   a) $25 \equiv 6 \pmod{m}$

   b) $17 \equiv 4 \pmod{m}$

   c) $4317 \equiv 7 \pmod{m}$

4. Bereken het laatste cijfer van het getal 2^{60}. Gebruik hiervoor modulo rekenen.

11.7 Oplossingen

1. a) Het maximum aantal is: $W_7^6 = 7^6 = 117\,649$.

   b) i. De gezochte code is 625316-2156.
   Werkwijze:
   $\sum_{i=1}^{6} c_i \equiv 6+2+5+3+1+6 \equiv 23 \equiv 2 \pmod 7$
   $\sum_{i=1}^{6} ic_i \equiv 6+4+1+5+5+1 \equiv 22 \equiv 1 \pmod 7$
   $\sum_{i=1}^{6} i^2c_i \equiv 6+1+3+6+4+6 \equiv 26 \equiv 5 \pmod 7$
   $\sum_{i=1}^{6} i^3c_i \equiv 1\cdot6+1\cdot2+6\cdot5+1\cdot3+6\cdot1+6\cdot6 \equiv 6+2+2+3+6+1 \equiv 20 \equiv 6 \pmod 7$

   ii. De gezochte code is $103526-3632$
   Werkwijze:
   $$\begin{array}{rcll} \sum_{i=1}^{6} i^2c_i \equiv 1+4x+6+3+1+6 & \equiv & 3 & \pmod 7 \\ & \Updownarrow & & \\ 4x+3 & \equiv & 3 & \pmod 7 \\ & \Updownarrow & & \\ 4x & \equiv & 3-3 \equiv 0 & \pmod 7 \\ & \Updownarrow & & \\ x & \equiv & 0\cdot4^{-1} \equiv 0 & \pmod 7 \end{array}$$
   De ontbrekende controle getallen zijn:

$$\sum_{i=1}^{6} c_i \equiv 1+0+3+5+2+6 \equiv 17 \equiv 3 \pmod 7$$
$$\sum_{i=1}^{6} ic_i \equiv 1+0+2+6+3+1 \equiv 13 \equiv 6 \pmod 7$$
$$\sum_{i=1}^{6} i^3c_i \equiv 1+0+4+5+5+1 \equiv 16 \equiv 2 \pmod 7$$

iii. De gezochte code is $452320 - 2001$
Werkwijze:

$$\begin{cases} \sum_{i=1}^{6} ic_i \equiv 1\cdot 4 + 2\cdot x + 3\cdot 2 + 4\cdot 3 + 5\cdot y + 6\cdot 0 \equiv 0 \pmod 7 \\ \sum_{i=1}^{6} i^2c_i \equiv 1\cdot 4 + 4\cdot x + 2\cdot 2 + 2\cdot 3 + 4\cdot y + 1\cdot 0 \equiv 0 \pmod 7 \end{cases}$$

$$\Leftrightarrow \begin{cases} 4+2x+6+5+5y+0 \equiv 2x+5y+1 \equiv 0 \pmod 7 \\ 4+4x+4+6+4y+0 \equiv 4x+4y \equiv 0 \pmod 7 \end{cases}$$

$$\Leftrightarrow \begin{cases} x \equiv (-4y)\cdot 4^{-1} \equiv 3\cdot y\cdot 2 \equiv 6y \pmod 7 \\ 12y+5y+1 \equiv 3y+1 \equiv 0 \pmod 7 \end{cases}$$

$$\Leftrightarrow \begin{cases} x \equiv 6\cdot 2 \equiv 5 \pmod 7 \\ y \equiv -1\cdot 3^{-1} \equiv 6\cdot 5 \equiv 2 \pmod 7 \end{cases}$$

De ontbrekende controle getallen zijn:

$$\sum_{i=1}^{6} c_i \equiv 4+5+2+3+2+0 \equiv 16 \equiv 2 \pmod 7$$
$$\sum_{i=1}^{6} i^3c_i \equiv 4+5+5+3+5+0 \equiv 22 \equiv 1 \pmod 7$$

iv. De gezochte code is $500000 - 5555$
Werkwijze:

$$\sum_{i=1}^{6} c_i \equiv x+0+y+u+v+0 \equiv x+y+u+v \equiv 5 \pmod 7$$
$$\sum_{i=1}^{6} ic_i \equiv 1\cdot x + 2\cdot 0 + 3\cdot y + 4\cdot u + 5\cdot v + 6\cdot 0 \equiv x+3y+4u+5v \equiv 5 \pmod 7$$
$$\sum_{i=1}^{6} i^2c_i \equiv 1\cdot x + 4\cdot 0 + 2\cdot y + 2\cdot u + 4\cdot v + 1\cdot 0 \equiv x+2y+2u+4v \equiv 5 \pmod 7$$
$$\sum_{i=1}^{6} i^3c_i \equiv 1\cdot x + 1\cdot 0 + 6\cdot y + 1\cdot u + 6\cdot v + 6\cdot 0 \equiv x+6y+u+6v \equiv 5 \pmod 7$$

De oplossing van dit stelsel kan onmiddellijk afgelezen worden uit de opgave. Wanneer $x \equiv 5$ en $y \equiv u \equiv v \equiv 0$ kloppen alle vergelijkingen.

Wanneer het niet voldoende duidelijk is op deze manier kan het stelsel in 4 vergelijkingen en 4 onbekenden steeds opgelost worden met bijvoorbeeld de methode van Gauss-Jordan.

2. a) $a \equiv 2 \pmod 3$.

   b) $a \equiv 5 \pmod{12}$ in het geval $\frac{a-5}{6}$ even is.
   $a \equiv 11 \pmod{12}$ in het geval $\frac{a-5}{6}$ oneven is.

3. a) $m = 19$.

   b) $m = 13$.

   c) $m = 10$.

4. Het laatste cijfer is: 6.
   Verklaring: de bewerking $\pmod{10}$ laten uitvoeren op een willekeurig getal $x \in \mathbb{N}$, heeft als resultaat de eenheden, d.i. het laatste cijfer, van x.

   $2^{60} \equiv (2^5)^{12} \equiv 32^{12} \equiv 2^{12} \equiv (2^6)^2 \equiv 64^2 \equiv 4^2 \equiv 16 \equiv 6 \pmod{10}$.

Bibliografie

[Biggs, 2006] N.L. Biggs, *Discrete Mathematics*, Oxford University press, 2006.

[Bóna, 2003] M. Bóna, *A walk through combinatorics*, World Scientific, 2003.

[Bouqué, 1967] E. Bouqué, *De algebra der verzamelingen en relaties*, story-scientia, 1967.

[Bouqué, 1972] E. Bouqué, *Boole'se algebra's*, Story-scientia, 1972.

[Bruyneel, 1999] G. Bruyneel, *Discrete Wiskunde*, cursus Hogeschool Gent, 1999.

[De Bock et al., 1989] D. De Bock, H. Eggermont, M. Roelens *Exponentiële en logaritmische functies*, Plantyn, 1989.

[De Pauw et al., 2009] I. De Pauw, B. Masselis, *Wiskunde voor multimedia*, Lannoo Campus, 2009.

[De Pauw et al., 2010] I. De Pauw, B. Masselis, *Wiskunde voor IT*, Lannoo Campus, 2010.

[De Smet, 1999] A. De Smet, *Wiskunde voor informatici*, cursus Hogeschool Gent, 1999.

[Hill, 1994] R. Hill, *A first course in coding theory*, Clarendon Press - Oxford, 1994.

[HoWest, 2007] Techno-ki(d)ts, *Codeertheorie, van bit tot code*, Hogeschool West-Vlaanderen - Wetenschap maakt knap, 2007.

[Maerivoet, 2004] R. Maerivoet, *Alice TI*, Standaard Boekhandel, 2004.

[Mattson, 1993] H.F. Mattson, *Discrete Mathematics with applications*, John Wiley & sons, inc. 1993.

[Opbouw, 1990] Opbouw 6B, *Boole-algebra's*, de sikkel, 1990.

[Papula, 1993] L. Papula, *Wiskunde voor het hoger technisch onderwijs*, Academic service, 1993.

[Tucker, 1995] A. Tucker, *Applied Combinatorics*, John Wiley & sons, inc. 1995.

[van As, 1996] D.P.G. van As, *Wiskunde voor de sector HEO*, Academic service, 1996.

[van den Hoek, 1997] C. van den Hoek, *HBO Wiskunde*, Academic service, 1997.

[van de Vrie et al., 1999] van de Vrie, Beerends, Haven, Lodder, Mulder, *Discrete Wiskunde*, Academic Service, 1999.

[Van Maldeghem, 2004] H. Van Maldeghem, *Pi, wiskunde, codeertheorie en cryptologie*, Uitgeverij de boeck, 2004.